Dimensioning cellular IoT network using stochastic geometry and machine learning techniques

Tuan Anh Nguyen

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2021IPPAT014

Thèse de doctorat

Dimensioning Cellular IoT network using Stochastic geometry and Machine learning techniques

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Télécom Paris

École doctorale n°626 École doctorale de l'Institut Polytechnique de Paris (EDIPP)
Spécialité de doctorat : Informatique et Réseaux

Thèse présentée et soutenue à Palaiseau, le 13 July 2021, par

NGUYEN TUAN ANH

Composition du Jury :

M. Laurent DECREUSEFOND Professeur, Télécom Paris, France	Président
M. Nadjib AIT SAADI Professeur, Université de Versailles Saint-Quentin-en-Yvelines, France	Rapporteur
M. Xuan Nam TRAN Professeur, Le Quy Don Technical University, Vietnam	Rapporteur
Mme Lina MROUEH Professeur, ISEP, France	Examinatrice
Mme Thi-Mai-Trang NGUYEN Mâıtre de Conférences HDR, Sorbonne Université, France	Examinatrice
M. Philippe MARTINS Professeur, Télécom Paris, France	Directeur de thèse

Acknowledgement

It is my pleasure to get the opportunity to express gratitude to all who contributed to the successful completion of my PhD thesis.

First, I would like to thank my supervisors, Prof. Philippe Martins for giving me the opportunity to carry out this thesis. I am very glad to work on this interesting topic and learn from them. His continuous support and guidance were valuable assets for the completion of this thesis. Especially, I would like to thank Prof. Van Tam Nguyen, Prof. Nguyen Thi Mai Trang and Dr. Germain Dang Kien for their effort and admirable dedication during this journey.

I am also grateful to the other members of my research group for their collaboration and fellowship. They are a wonderful group of people able to achieve incredible milestones. Special thanks to Sebastien and Aymeric for their time, for their technical help and for inspiring me.

Finally, I want to thank my family for their unconditional love and support from the other side of the world. Most of all, I am tremendously grateful for the continuous support and love I received from my wife and son, and their patience during the vicissitudes of my peculiar mind in this thesis.

Contents

Résumé étendu

Introduction

Les réseaux mobiles ont eu différents impacts sur notre vie scolaire, comme nos réseaux professionnels, les interactions sociales, la communication, etc. Les communications mobiles sans fil ont connu une énorme progression, passant d'une technologie sans fil initiale coûteuse à la popularité actuelle. Jusqu'à présent, quatre générations de réseaux cellulaires, de la 1G à la 4G, ont été déployées, et la cinquième génération (5G) a été introduite et appliquée dans certains pays. De nouveaux cas d'utilisation apparaissent à chaque expansion des réseaux cellulaires. Les générations 1G et 2G étaient des systèmes centrés sur la voix. Plus tard, la 3G et la 4G ont également pris en charge les services de données par paquets. Chaque génération ultérieure améliore les avantages de la génération précédente et apporte une évolution technologique significative. Contrairement aux quatre premières générations qui fournissent davantage de services aux utilisateurs humains, la 5G permet des solutions innovantes pour les machines et les humains.

Ces dernières années, l'Internet des objets (IoT) a été un sujet brûlant, tant dans les domaines de la recherche que sur le marché industriel. La volonté de connecter à internet tout ce qui existe, comme les appareils, les machines et les véhicules, est à l'origine de cette tendance. Cisco estime que, d'ici 2023, il y aura 15 milliards de dispositifs communicants dans le monde [1]. Ces dispositifs peuvent être d'intérieur ou d'extérieur, de quelques octets par jour pour les compteurs intelligents à une énorme quantité d'informations par milliseconde pour les moniteurs de véhicules, de Bluetooth ou de WiFi aux technologies sub-GHz, et de batteries comme les dispositifs portables à une charge permanente comme les caméras de surveillance. Avec des caractéristiques différentes, leurs exigences particulières varient.

Dans les réseaux IoT, la plupart des appareils ne sont pas très exigeants en termes de latence et de débit de données. Ils n'ont qu'une petite quantité de données à transférer et ne nécessitent pas une réponse avec un délai de quelques millisecondes. Aujourd'hui, ces appareils sont généralement connectés par le biais de la norme 802.11 ; cependant, ces communications ont une portée limitée. Du point de vue de la capacité, un eNB peut desservir une forte densité de dispositifs avec une couverture de plusieurs dizaines de kilomètres. Cependant, pour éviter l'entretien fréquent des batteries d'un grand nombre de dispositifs, la faible consommation d'énergie est une exigence forte de l'IdO. Par conséquent, les réseaux IoT doivent avoir une large couverture, être capables de gérer un grand nombre de dispositifs et maintenir une faible consommation d'énergie ainsi qu'un faible coût des dispositifs.

Afin de répondre aux exigences du monde de l'IoT, de récentes technologies dédiées aux réseaux étendus à faible consommation d'énergie (LPWAN) ont fait leur apparition, comme NB-IoT, qui a été inclus dans la normalisation 3GPP, et qui est le système à bande étroite à faible coût et faible consommation d'énergie basé sur LTE ; LoRa, qui donne la possibilité aux

gens de construire leur propre réseau LoRaWAN ; et UNB (Ultra Narrow Band), développé et breveté par la société française SigFox, qui a été le premier à être lancé commercialement pour l'IoT.

Dans cette thèse, nous nous concentrons sur le 3GPP NB-IoT. Cette solution devrait élargir le champ des applications 5G pour inclure les villes intelligentes, le suivi des objets et le contrôle à distance, entre autres. NB-IoT peut fonctionner dans un spectre aussi étroit que 200 kHz en réformant les porteuses GSM et LTE et en réutilisant les infrastructures de réseau, ce qui représente un avantage majeur pour les opérateurs tout en étant capable de fonctionner dans le spectre LTE. Les principaux objectifs de conception du NB-IoT sont l'augmentation de la couverture intérieure, la prise en charge d'un nombre massif de connexions (jusqu'à 50000 par cellule de réseau NB-IoT), une longue durée de vie des batteries (jusqu'à 10 ans) et une faible complexité des dispositifs d'équipement des utilisateurs. NB-IoT est très flexible et peut fonctionner dans n'importe quelle bande 2G/3G/4G (de 450 Mhz à 3-5 GHz) puisqu'il atteint une excellente performance de coexistence et de compatibilité avec les systèmes cellulaires existants. NB-IoT offre une plus grande fiabilité et une meilleure qualité de service (QoS) car il fonctionne dans une petite partie du spectre cellulaire disponible existant. En outre, NB-IoT peut utiliser l'infrastructure de réseau cellulaire existante, ce qui réduit les coûts de déploiement.

L'IoT à bande étroite peut offir les meilleures technologies dans les villes intelligentes et les projets agricoles. Par exemple, dans les villes intelligentes, la technologie NB-IoT peut surveiller la pollution atmosphérique, le traffic, les compteurs intelligents et les processus qui nécessitent l'envoi peu fréquent de petits paquets de données au centre du réseau ou à une plateforme de gestion IoT. Comme ces dispositifs peuvent être situés en profondeur à l'intérieur, voire sous terre, cela réduit la couverture en raison de l'atténuation introduite par les bâtiments. En outre, dans les zones urbaines denses, un grand nombre de ces appareils utilisent généralement la même cellule, ce qui pose le problème de la gestion des ressources radio.

Il est donc important de bien dimensionner le réseau, en tenant compte du nombre de dispositifs, de leur densité et de leur emplacement, pour obtenir les meilleures performances. Sinon, nous risquons d'avoir un réseau incapable de fournir les services ou même de perturber les services déjà mis en œuvre en surchargeant les ressources radio dans une cellule, ce qui rendrait les communications inopérantes. En outre, dans les réseaux modernes récents avec des ressources radio à fréquence temporelle limitées et très coûteuses, la classification et la connaissance proactive du traffic entrant peuvent améliorer de manière significative l'utilisation des ressources. Elle permet au système une allocation optimale des ressources. Dans ce contexte, les principales préoccupations lors du dimensionnement des réseaux NB-IoT sont la couverture (c'est-à-dire le nombre de dispositifs connectés par unité de surface), la gestion des ressources (c'est-à-dire le nombre d'allocations de ressources nécessaires pour garantir la qualité de service du réseau), et la reconnaissance et la prédiction du traffic (pour la prédiction de la distribution du traffic et la prédiction du comportement futur des utilisateurs). À cette fin, nous visons à utiliser des approches de géométrie stochastique et d'apprentissage automatique pour étudier le dimensionnement du réseau NB-IoT.

Évaluation des performances de couverture de la transmission de la liaison montante dans les réseaux NB-IoT.

La principale contribution de ce travail est la dérivation de la probabilité de couverture de la liaison montante dans les réseaux NB-IoT à cellule unique. L'emplacement des capteurs est modélisé comme une réalisation de PPP et la station de base est située à l'origine. Sous ces hypothèses, les expressions analytiques des probabilités de couverture et d'accès réussi sont dérivées en prenant en compte la distribution d'arrivée des paquets.

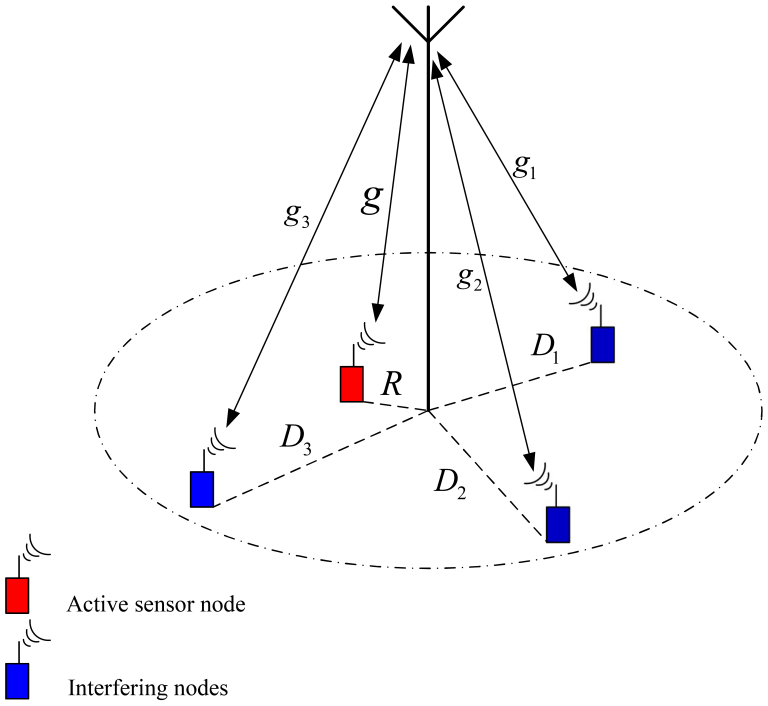


Figure 1: Modèle de système de liaison montante NB-IoT.

La figure 1 illustre le modèle du système de liaison montante et la relation entre les paramètres. Nous considérons un système à bruit limité où les emplacements des nœuds sont supposés former une réalisation d'un PPP homogène de densité λ_n.

La distance R entre le dispositif NB-IoT et l'eNB est supposée être une variable aléatoire i.i.d. suivant la distribution de Rayleigh avec la fonction de densité de probabilité (pdf) de R suivante :

$$f_R(r) = 2\pi\lambda r e^{-\lambda\pi r^2}, r \geq 0 \tag{1}$$

Dans ce modèle de système, le SINR d'un eNB situé à l'origine est :

$$\text{SINR} = \frac{gpR^{-\alpha}}{\sigma^2 + I_Z}. \tag{2}$$

où σ^2 modélise la puissance du bruit thermique et I_Z désigne l'interférence créée par l'ensemble des dispositifs interférents Z :

$$I_Z = \sum_{z_i \in Z} (p g_i D_i^{-\alpha}) \tag{3}$$

Supposons que les tentatives de transmission du canal à l'eNB suivent une distribution de Poisson et que la probabilité de sélectionner un préambule suit une distribution uniforme. Une collision se produit lorsque plus d'un capteur transmet au même moment avec le même préambule. Sur la base de la distribution de Poisson, la probabilité de collision p_f est alors égale :

$$p_f = 1 - \frac{\sum_{i=1} i \frac{(\lambda_{Total}/N_{ra})^i}{i!} \exp(-\lambda_{Total}/N_{ra})}{\lambda_{Total}/N_{ra}} \tag{4}$$

$$= 1 - \exp\left(-\frac{\lambda_{Total}}{N_{ra}}\right) = 1 - p_s, \tag{5}$$

où λ_{Total} est le taux d'arrivée total, λ est l'intensité des nouvelles arrivées, N_{ra} est le nombre de préambules d'accès aléatoire et p_s est la probabilité d'accès réussi.

Chaque paquet subit la probabilité d'interruption du canal, qui est définie comme la probabilité que le SINR à l'entrée du récepteur tombe en dessous d'une valeur seuil donnée T. Mathématiquement, la probabilité d'interruption peut être obtenue à partir de la CDF du SINR et est donnée par $P\{SINR \leq T\}$. De là, la probabilité d'accès réussi est :

$$p_s = e^{-\lambda_{Total}/N_{ra}} . \left(1 - P\{SINR < T\}^{N_r}\right)$$
$$= e^{-\lambda_{Total}/N_{ra}} . (1 - (1 - p_c)^{N_r}), \tag{6}$$

où N_r est le nombre de répétitions et $p_c = P\{SINR \geq T\}$ est la probabilité de couverture.

En utilisant la fonction Lambert W, le taux d'arrivée total peut être calculé comme suit

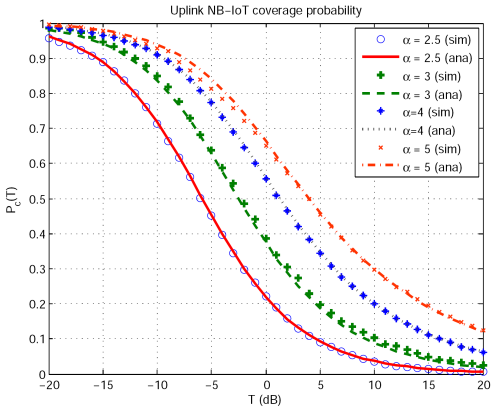
$$\lambda_{Total} = -N_{ra} . W_0 . \left(-\lambda/(N_{ra} . Q)\right), \tag{7}$$

où W_0 indique la fonction Lambert W.

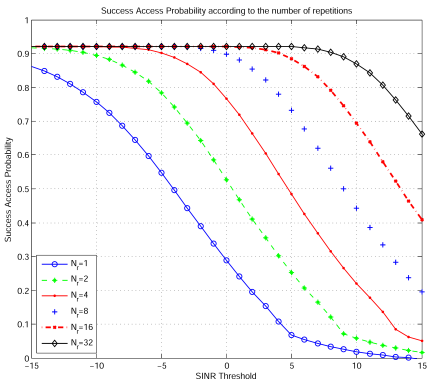
Le brouillage au niveau de l'eNB est la somme des puissances de tous les dispositifs brouilleurs (modélisés par le PPP). Cette puissance dépend de la distance entre le dispositif et l'eNB. En utilisant la fonction génératrice de moment et la fonction génératrice de probabilité, on obtient la transformée de Laplace de la distribution des interférences observées au niveau de l'eNB:

$$L_{I_Z}(s) = exp\left[-2\pi\lambda_n \int_r^\infty \left(1 - \left[\frac{1}{1 + (sp)x^{-\alpha}}\right]\right) x \mathrm{d}x\right]$$
$$= exp\left[-2\pi\lambda_n \int_r^\infty \left[\frac{1}{1 + (sp)^{-1}x^{\alpha}}\right] x \mathrm{d}x\right] \tag{8}$$

La probabilité de couverture de la liaison montante est définie comme la fonction de distribution cumulative complémentaire (CCDF) du SINR de la liaison montante, c'est-à-dire la



(a) Probabilité de couverture NB-IoT sur la liaison montante.



(b) Probabilité d'accès réussi dans la liaison montante NB-IoT.

Figure 2: Simulation et analyse des performances.

probabilité que le SINR de la liaison montante au niveau de l'eNB marqué soit supérieur au SINR cible (T):

$$\begin{aligned} p_c(T, \lambda_n, \alpha) &= \int_0^\infty P\{SINR \geq T\} f_R(r) \mathrm{d}r \\ &= \int_0^\infty 2\pi\lambda_n r e^{-\pi\lambda_n r^2 - Tp^{-1}r^\alpha \sigma^2} . L_{I_Z}\left(Tp^{-1}r^\alpha\right) \mathrm{d}r \end{aligned} \tag{9}$$

Et nous avons la probabilité d'un accès réussi:

$$p_s = e^{-\lambda_T / N_{ra}} \int_0^\infty 2\pi\lambda_n r e^{-\pi\lambda_n r^2 - Tp^{-1}r^\alpha \sigma^2} . L_{I_Z}\left(Tp^{-1}r^\alpha\right) \mathrm{d}r \tag{10}$$

Nous évaluons les performances d'un réseau NB-IoT en liaison montante et validons notre modèle analytique pour la probabilité de couverture à l'aide de simulations de Monte-Carlo. La figure 2a présente la probabilité de couverture de la liaison montante NB-IoT en fonction du seuil SINR pour les valeurs de l'exposant de l'affaiblissement du parcours de $\alpha = 2, 5, 3, 4, 5$.

La figure 2b représente la probabilité d'accès réussi en fonction du seuil SINR pour différents nombres de répétitions. La probabilité d'accès réussi est améliorée lorsque N_r augmente en raison de l'augmentation de la probabilité de couverture. Si nous supposons que le seuil SINR est de -5 dB, la probabilité d'accès réussi peut être considérée comme étant d'environ 0.9 lorsque $N_r = 4$.

Ce travail a présenté des expressions pour la probabilité de couverture et la probabilité d'accès réussi dans la liaison montante NB-IoT. Les expressions sont basées sur un modèle analytique utilisant la géométrie stochastique et sont fonction des paramètres de conception du système, notamment la densité des dispositifs de détection et les objectifs de SINR. Les simulations montrent que le modèle proposé est précis.

Dimensionnement des liaisons descendantes multicellulaires dans les réseaux NB-IoT

La principale contribution de notre travail est la dérivation d'un modèle numérique pour calculer les blocs de ressources radio nécessaires dans le réseau NB-IoT descendant multi-cellules en utilisant les polynômes de Bell. Ce modèle est très important pour les opérateurs car il montre comment gérer les ressources spectrales disponibles. Sur la base du nombre de blocs de ressources requis, nous déterminons la probabilité d'interruption du réseau en fonction du taux de transmission des capteurs, de l'atténuation des pertes sur le trajet et des densités des dispositifs IoT et des eNB.

Nous considérons la performance du canal descendant dans le réseau NB-IoT. Les réseaux sont déployés selon un processus ponctuel de Poisson. Comme nous étudions un système NB-IoT, le canal descendant de chaque antenne est divisé en sous-porteuses de 15 kHz. Supposons que chaque cellule utilise une bande de fréquence différente de celle de ses voisines et dispose de M blocs de ressources disponibles. Ces blocs de ressources sont ensuite attribués aux utilisateurs actifs, dont les positions sont tirées selon un processus ponctuel de Poisson. Les positions des utilisateurs actifs dans la cellule suivent un PPP spatial Φ_u d'intensité λ_u. Dans une zone de couverture donnée, le nombre de capteurs suit une distribution de Poisson tandis que leurs emplacements suivent une distribution uniforme.

Nous supposons que chaque utilisateur est connecté à l'eNB le plus proche. Les limites des cellules forment une tessellation de Voronoï sur le plan, comme l'illustre la figure 3.

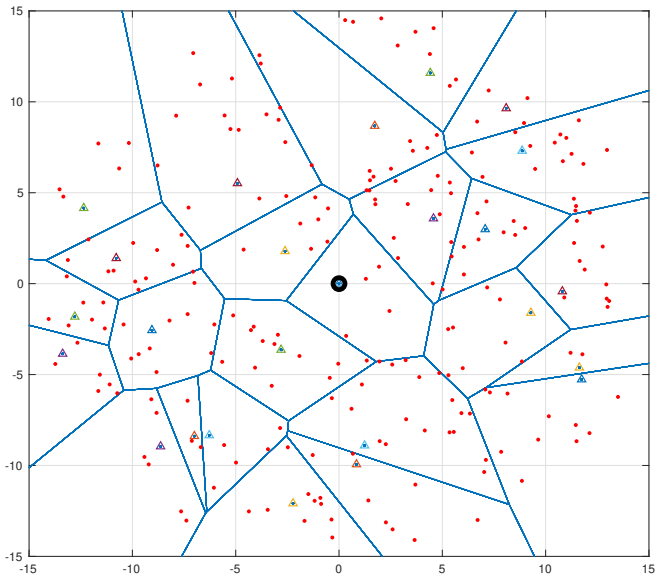


Figure 3: Une réalisation du réseau modélisé par un PPP Φ_b et Φ_u. L'eNB typique est représentée par un point noir alors que les capteurs sont représentés par des petits points rouges.

En fonction du taux de transmission et de la position du capteur, chaque utilisateur dispose

d'un nombre donné de n PRBs. Dans ce travail, nous désignons le taux de transmission requis par C^*.

L'utilisateur du capteur situé à une distance x de O ne peut décoder le signal que si le rapport signal/bruit (SINR) $\text{SINR}(x) = \frac{P_t x^{-\alpha}/a}{I+\sigma^2}$ est supérieur à un seuil θ_{thr}, où I est l'interférence cocanal reçue et est supposé négligeable dans notre analyse. σ^2 est la puissance du bruit thermique. Si le SINR est inférieur au seuil critique, on dit que le capteur est en panne et qu'il ne peut pas poursuivre sa communication.

L'eNB peut allouer un nombre maximal N_{max} de blocs de ressources à chaque nœud de capteur à chaque intervalle de temps. Selon la formule de Shannon, un capteur situé à une distance x du centre O demandant un service à un débit de données C^*, a le nombre de PRBs qui est défini comme suit :

$$\begin{aligned} N(x) &= \min\left(\left\lceil \frac{C^*}{W_{rb} log_2(1+\text{SINR}(x))} \right\rceil, N_{max}\right) \\ &= \min\left(\left\lceil \frac{C^*}{C(x)} \right\rceil, N_{max}\right) \end{aligned} \tag{11}$$

où $\lceil y \rceil$ est la fonction de plafonnement de y, W_{rb} est la largeur de bande d'un bloc de ressources, la quantité N_{max} est le nombre maximal de blocs de ressources qui peuvent être alloués à un utilisateur de capteur et $C(x)$ est le débit d'un capteur situé à une distance x de O.

D'après (11), on se rend compte que le capteur dans lequel la qualité du canal est mauvaise a besoin d'un plus grand nombre de PRBs pour atteindre le taux de transmission C^*. Soit d_n la distance de O qui vérifie, pour tout $x \in (d_{n-1}, d_n]$, $N(x) = n$, avec

$$n = \frac{C^*}{C(d_n)} \tag{12}$$

est un nombre entier et

$$d_n = \begin{cases} 0 & \text{if } n = 0 \\ \left[\frac{a(I+\sigma^2)}{P}\left(2^{\frac{C^*}{nW_{rb}}} - 1\right)\right]^{-\frac{1}{\alpha}} & \text{otherwise.} \end{cases} \tag{13}$$

D'après (12), la zone de la cellule type O peut être divisée en anneaux de rayon d_n ($1 \leq n \leq N_{max}$, $0 \leq d_{n-1} \leq d_n$). Lorsque le capteur demande n PRBs pour atteindre le taux de transmission C^*, on dit que le capteur est situé dans la zone entre deux anneaux de rayon d_n et d_{n-1}. Si $x > d_{N_{max}}$, on considère que le capteur est hors service et $n = 0$.

De plus, on se rend compte que le nombre de répétitions de chaque utilisateur dépend de sa valeur SINR. Soit r_l la distance de O, la cellule typique peut être divisée en anneaux de rayon r_l ($0 \leq l \leq 7, 0 \leq r_{l-1} \leq r_l$). La zone comprise entre les anneaux de rayon r_l et r_{l-1} définit la région de la cellule où les capteurs ont $n_{rep}(l)$ répétitions ($n_{rep}(l) = 2^l$, $l = 0, 1, .., 7$). Pour chaque utilisateur ayant une distance $x \in (r_{l-1}, r_l]$, il a le nombre de répétitions $n_{rep}(l)$ avec :

$$r_l = \begin{cases} 0 & \text{if } n = 0 \\ \left[\frac{a(I+\sigma^2)}{P}\text{SINR}_{\text{target}}(l)\right]^{-\frac{1}{\alpha}} & \text{otherwise.} \end{cases} \tag{14}$$

Le nombre total de PRB requis dans la cellule type est défini comme la somme des PRB demandés par les utilisateurs des capteurs dans chaque anneau de rayon d_n et peut être représenté comme suit :

$$\Gamma = \sum_{n=1}^{N_{max}} nV_n \tag{15}$$

où le nombre d'utilisateurs qui se trouvent entre deux anneaux $B(0, d_n)$ et $B(0, d_{n-1})$ est une variable aléatoire de Poisson désignée par V_n :

$$V_n = \sum_{l=0}^{7} X_n^{n_{rep}(l)} \tag{16}$$

La probabilité de panne est définie comme la probabilité que le nombre total de PRB demandés dans la cellule soit supérieur à une valeur seuil fixée par l'opérateur : $P_{out} = P(\Gamma \geq M)$, où M est le nombre de PRB de sortie requis pour garantir une qualité de services prédéfinie.

$$\mathbb{P}(\Gamma \geq M) = \mathbb{E}_{\Phi_b}[\mathbb{P}(\Gamma \geq M|\Phi_b)] \tag{17}$$

Cette formule peut être encore améliorée en introduisant un modèle mathématique appelé polynômes de Bell. Cela nous permet de déterminer la probabilité de panne comme une somme finie des polynômes de Bell :

Soit Λ une variable aléatoire telle que $\Lambda = \sum_{n=1}^{N} nV_n$, avec V_n sont des variables aléatoires de Poisson d'intensité w_n. Soit x_j défini comme

$$x_j = \begin{cases} w_j j! & \text{if } 1 \leq j \leq N \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

La probabilité que Λ dépasse un seuil M peut être exprimée comme une fonction des polynômes de Bell complets exponentiels par

$$\mathbb{P}(\Lambda \geq M) = 1 - H \sum_{k=0}^{M-1} \frac{B_k(x_1, ..., x_k)}{k!} \tag{19}$$

où $H = e^{-\sum_{n=1}^{N} w_n}$.

La formule analytique de la probabilité de panne est ensuite comparée aux résultats des simulations de Monte-Carlo, comme le montrent les figures suivantes :

La figure 4 illustre le modèle décrit dans MATLAB pour la densité des eNB $\lambda_b = 0,9\, eNBs/km^2$ et deux valeurs de la densité des utilisateurs de capteurs actifs $\lambda_u = 15\ utilisateurs/km^2$, $\lambda_u = 20\ utilisateurs/km^2$. Nous observons que le modèle analytique de la probabilité de panne correspond au résultat de la simulation de Monte-Carlo. De plus, l'augmentation de l'intensité des capteurs actifs génère une augmentation de la probabilité de panne. En d'autres termes, le système connaît une forte probabilité d'interruption lorsque le nombre de PRB requis par les utilisateurs augmente, en raison de l'intensité croissante.

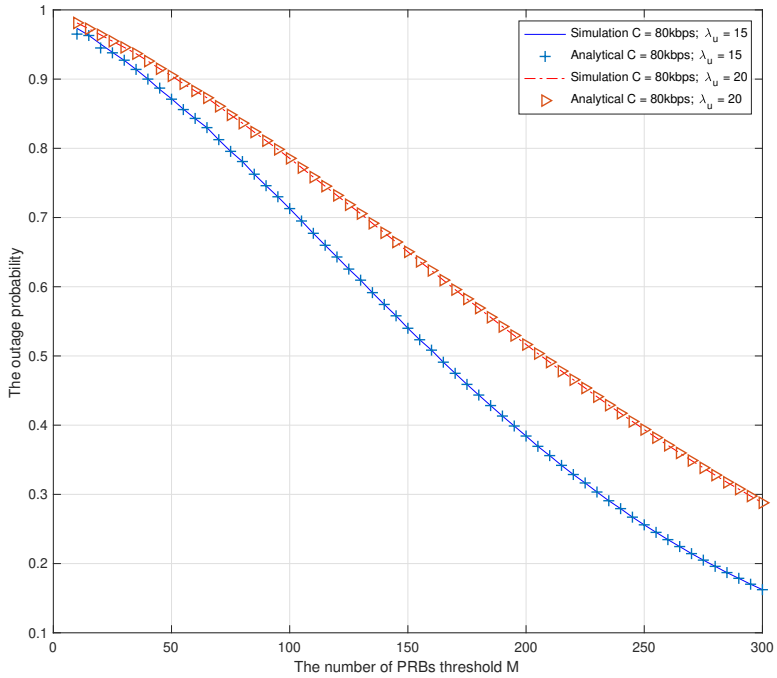
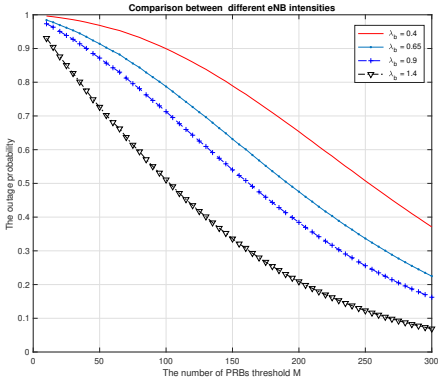


Figure 4: La probabilité de panne dans l'analyse et la simulation.

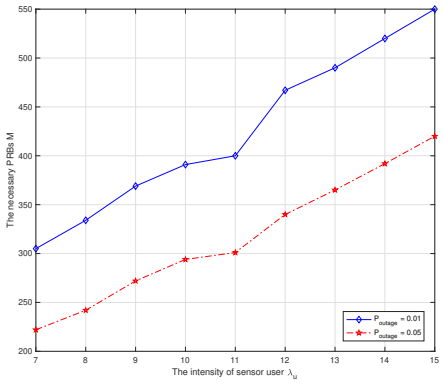
Pour voir l'impact de l'intensité de l'eNB sur les performances, nous traçons la probabilité d'interruption, en considérant différentes valeurs d'intensité de l'eNB tout en conservant l'intensité du capteur actif $\lambda_u = 15$ $utilisateurs/km^2$ dans la Figure 5a. Nous observons que la probabilité de panne diminue lorsque l'intensité de l'eNB augmente. En d'autres termes, si le nombre moyen d'utilisateurs de capteurs qu'une eNB doit servir est le même, l'augmentation du nombre d'eNB entraîne une diminution de la PRB nécessaire dans chaque eNB.

Pendant la procédure de dimensionnement des ressources, la BNE commence par définir une probabilité d'interruption cible qui peut être tolérée pour un service donné. Pour un débit de transmission donné, le nombre de PRB est fixé de manière à garantir que la probabilité de panne ne dépasse jamais sa valeur cible. La figure 5b présente le nombre de PRB nécessaires pour garantir la valeur cible de la probabilité d'interruption ($P_{out} = 1\%$ et $P_{out} = 5\%$) avec un débit de transmission fixe de $80kbps$. Nous pouvons observer que pour la valeur cible de la probabilité de panne, le nombre de blocs de ressources requis dans la cellule augmente lorsque l'intensité des utilisateurs de capteurs augmente.

La principale contribution de ce travail est la dérivation d'un modèle numérique pour calculer les blocs de ressources radio nécessaires dans le réseau NB-IoT descendant multi-cellules en utilisant les polynômes de Bell. Ce modèle est très nécessaire pour les opérateurs car il montre comment gérer la ressource spectrale disponible. Sur la base du nombre de blocs de ressources requis, nous déterminons la probabilité d'interruption du réseau en fonction du taux de transmission des capteurs, de l'atténuation de la perte de chemin et des densités des dispositifs IoT et des eNB.



(a) Comparaison entre différentes intensités d'eNB.



(b) PRB requis M en fonction de l'intensité du capteur.

Figure 5: Simulation et analyse des performances.

Reconnaissance et prédiction des types de trafic cellulaire

La reconnaissance et la prédiction des modèles de trafic cellulaire 4G et 5G sont des objectifs clés pour l'optimisation des réseaux. Elles deviennent également d'une importance fondamentale pour le réseau cellulaire de prochaine génération. La reconnaissance des modèles de trafic mobile et la connaissance proactive des comportements des utilisateurs permettent à l'opérateur d'optimiser l'allocation des ressources. D'un autre côté, il s'agit d'un problème complexe en raison de la diversité des applications générant le trafic. La plupart des problèmes de prédiction du trafic se concentrent sur la capture de la dynamique du trafic et l'amélioration des performances. Dans ce travail, nous concevons un modèle d'apprentissage profond pour la reconnaissance des modèles de trafic et la prédiction du type de paquet d'arrivée en utilisant des réseaux neuronaux à mémoire à long terme (LSTM). Les informations sur le trafic mobile sont collectées à partir des informations de contrôle de la liaison descendante (DCI) à l'aide du logiciel Amarisoft. La phase d'apprentissage du modèle s'appuie sur un modèle de trafic bien connu simulé sur le banc d'essai 4G et 5G d'Amarisoft.

Dans ce travail, nous nous intéressons à l'étude du trafic échangé entre l'eNodeB et tous les utilisateurs. Supposons que $\{w_1, ..., w_K\}$ sont les étiquettes des modèles de trafic. Les algorithmes proposés visent à apprendre une fonction $F(\mathbf{x}) = w_c \in \{w_1, ..., w_L\}$, où $\mathbf{x}$ est le vecteur de caractéristiques du flux de dimension D, c'est-à-dire $\mathbf{x} = (x_1, x_2, .., x_D)$. Soit T la période de mesure totale, pour chaque seconde $t \in T$, nous avons défini $\mathbf{x}(t)$ comme le vecteur qui contient l'information suivante :

- DL_{br} : Débit binaire de la liaison descendante (DL) en bits par seconde ;
- RB_{DL} : Nombre moyen de blocs de ressources DL alloués ;
- UL_{br} : Débit de la liaison montante (UL) en bits par seconde ;
- RB_{UL} : Nombre moyen de blocs de ressources UL alloués.

La séquence $\mathbf{X} = [\mathbf{x}(t)]_{t=1}^{T}$ est une série temporelle multi-variable, qui est recueillie à partir du message DCI via l'interface Amarisoft.

Le processus d'apprentissage prend en entrée un jeu de données $\mathcal{D} = \{(\mathbf{x}(i), y(i)), i = 1, ..., N\}$, où $\mathbf{x}(i)$ et $y(i)$ sont respectivement le vecteur de caractéristiques et l'étiquette du i-ième flux. Sur la base de $\mathcal{D}$, l'objectif est capable de reconnaître les échantillons des classes connues (c'est-à-dire $w_1, ..., w_K$) et de prédire le type de trafic au prochain pas de temps.

La figure 6 montre le modèle d'apprentissage profond proposé, qui comprend trois parties principales : le prétraitement des données, le reconnaissant et le prédicteur. La partie prétraitement comprend le fenêtrage des données, le scaler, le One-hot-Encoder et l'Autoencoder, tandis que les deuxième et troisième parties font référence à la formation du recognizer et du predictor basés sur LSTM, respectivement.

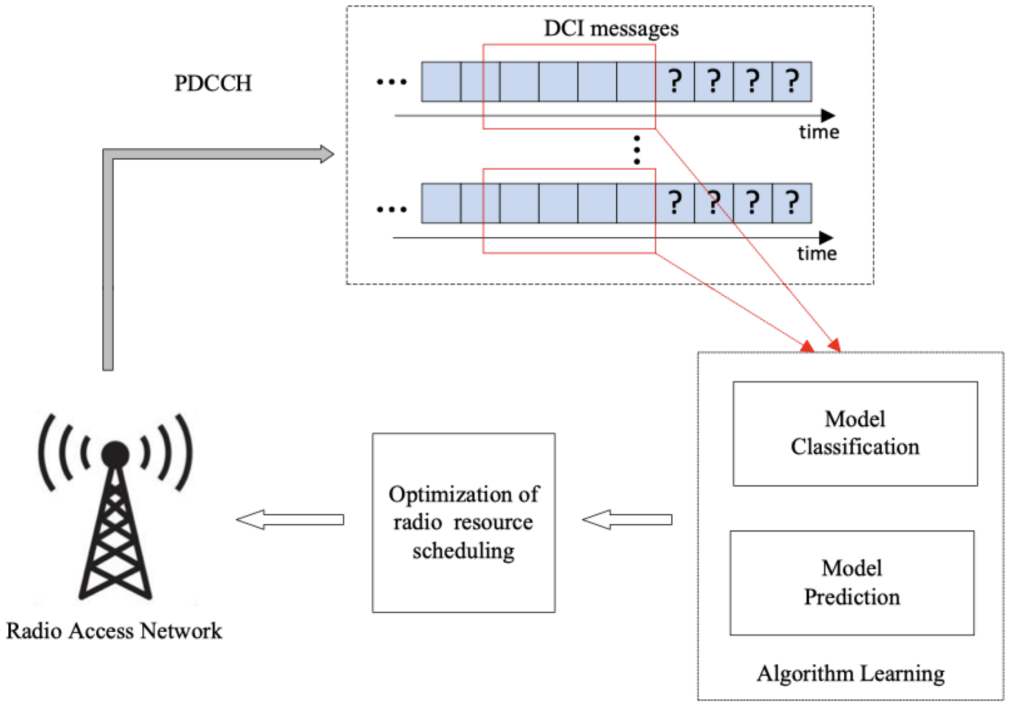


Figure 6: Le modèle d'apprentissage profond proposé.

Le Recognizer

Un reconnaissant estime une fonction $\Phi_c : \mathbf{X} \rightarrow Y$, où la matrice $\mathbf{Y}$ a une taille $N \times K$ et K représente le nombre de classes. La matrice $\mathbf{Y}$ est créée après une étape de codage à un coup. Le vecteur ligne $\Phi_c(\mathbf{x}(n)) = \hat{y}_c(n + W - 1) = [y_1, \cdots, y_K]$ avec $\sum_k y_k = 1$, contient les probabilités que les données de l'horodatage $n + W - 1$ appartiennent à chacune des K classes. On dit que la classe k^* est la sortie finale du classificateur lorsque $k^* = \text{argmax}_k(y_k)$.

L'architecture proposée pour la reconnaissance du type de trafic mobile est décrite dans la figure 7. Dans cette conception, nous considérons plusieurs couches d'unités LSTM pour former un réseau LSTM empilé.

The predictor

Nous utilisons une architecture many-to-one pour la prédiction en une seule étape, ce qui signifie que le réseau observe le trafic mobile pendant une durée W jusqu'à T, puis tente de prédire le type de trafic dans le créneau temporel suivant $T + 1$. La figure 8

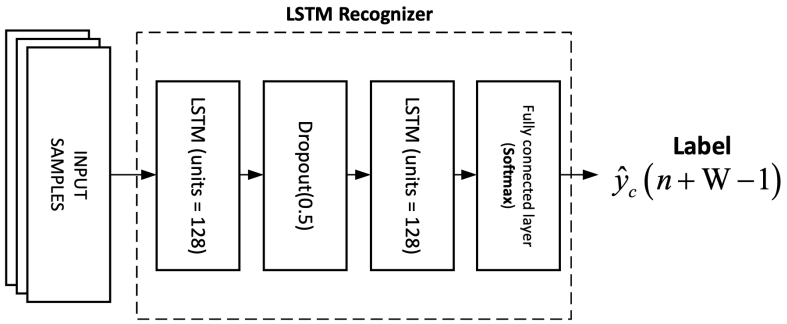


Figure 7: Le modèle de reconnaissance.

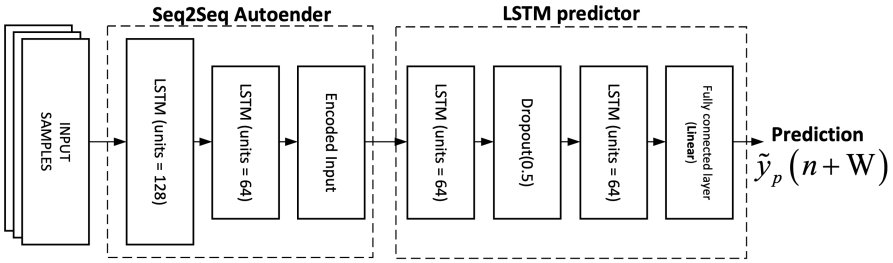


Figure 8: Le modèle de prédiction.

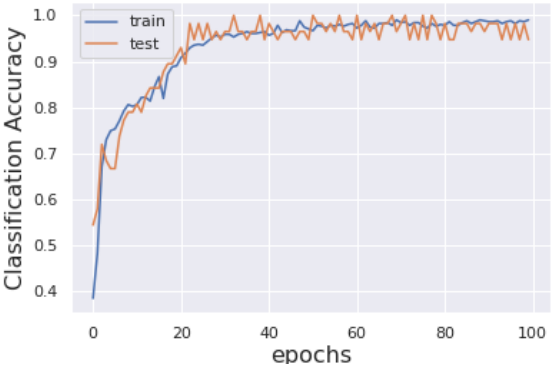
présente l'architecture LSTM pour effectuer la prédiction du type de trafic. Le modèle que nous proposons consiste en une architecture empilée qui comprend plusieurs couches LSTM. L'algorithme de prédiction Φ_p est enseigné pour prédire le type de trafic pour lequel l'erreur de prédiction devrait être faible. L'algorithme proposé reçoit une séquence de trafic de longueur W : $\tilde{\mathbf{x}}(n) = [\mathbf{x}(n), \mathbf{x}(n+1), ..., \mathbf{x}(n+W-1)]$, et tente de prédire le type de trafic au temps $n+W$, $\tilde{y}(n+W)$.

$$\Phi_p\left(\tilde{\mathbf{x}}(n)\right) = \tilde{y}_p(n+W) \tag{20}$$

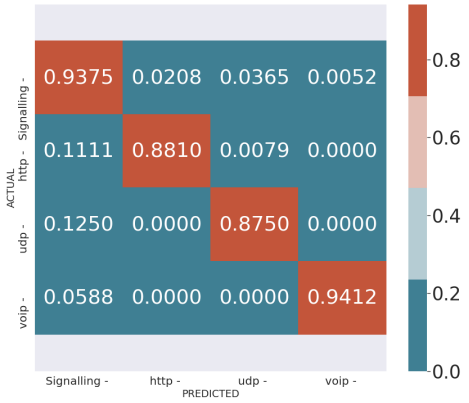
La figure 9a illustre la précision de reconnaissance du modèle proposé en fonction du nombre d'époques. Les performances s'améliorent après chaque époque. La précision obtenue est supérieure à 90% après 20 époques. On peut observer qu'il est possible d'atteindre un niveau de précision allant jusqu'à $98,25\%$ pour la tâche de reconnaissance.

La matrice de confusion avec normalisation par la taille du support de classe (nombre d'éléments dans chaque classe) est fournie dans la figure 9b pour analyser quels types sont mal reconnus dans le processus de reconnaissance. Il y a toujours des reconnaissances erronées entre les types de trafic Signalisation, HTTP et UDP. En fait, 11% et $12,5\%$ du HTTP et de l'UDP sont reconnus à tort comme de la signalisation, tandis que 2% et $3,6\%$ de la signalisation sont classés à tort comme du HTTP et de l'UDP, respectivement. Les résultats obtenus confirment en outre que des "modèles" similaires de types de traffic affectent les performances de reconnaissance, en particulier avec les trafics HTTP et UDP.

La performance de la prédiction du type de paquet arrivant dans un slot temporel est présentée dans la Figure 10. Les pertes d'apprentissage et de test après chaque époque sont données pour montrer la vitesse de convergence du modèle proposé. Il est possible d'observer



(a) La précision de la tâche de reconnaissance.



(b) Matrice de confusion dans la tâche de reconnaissance.

Figure 9: La performance de la reconnaissance

que la perte diminue rapidement pendant les 30 premières époques, puis atteint une faible perte de performance de $0,08$ après 90 époques. Les résultats obtenus montrent la capacité du LSTM à traiter convenablement les données de séries temporelles pour la prédiction du type de trafic.

L'avantage d'appliquer l'autoencodeur dans le modèle de prédiction est présenté en considérant deux matrices de confusion dans la figure 11. En utilisant un autoencodeur, l'approche de prédiction proposée peut clairement classer les types de trafic. En fait, $35,21\%$ du trafic UDP est prédit à tort comme étant de la signalisation. Ce pourcentage diminue à 15.49% en utilisant l'autoencodeur. En outre, dans ce cas, il est possible d'observer comment l'approche autoencodeur fournit de meilleurs résultats en ce qui concerne ces mesures pour la prédiction simple sans approche autoencodeur.

Notez qu'il existe de nombreux travaux utilisant les LSTM pour prévoir les données de séries temporelles. Mais la plupart d'entre eux se concentrent sur la capture de la dynamique des séries temporelles ou sur la prédiction des demandes futures. Notre travail se concentre sur la reconnaissance des modèles de trafic cellulaire, ce qui n'est pas le même problème que les travaux précédents.

Ce travail a dérivé un modèle d'apprentissage profond basé sur les réseaux LSTM pour la reconnaissance et la prédiction du type de trafic mobile, qui utilise les données collectées à partir des informations de contrôle de la liaison descendante et l'application d'un autoencodeur (pour la tâche de prédiction). L'analyse a montré qu'en utilisant un réseau LSTM, nous pouvons obtenir les meilleurs résultats avec une précision de plus de 98% pour la reconnaissance et moins de 10^{-1} pour la prédiction. Une comparaison avec l'utilisation d'un auto-codeur et sans auto-codeur a également démontré que le modèle de prédiction avec auto-codeur offre toujours une garantie de performance supérieure à l'approche d'apprentissage simple.

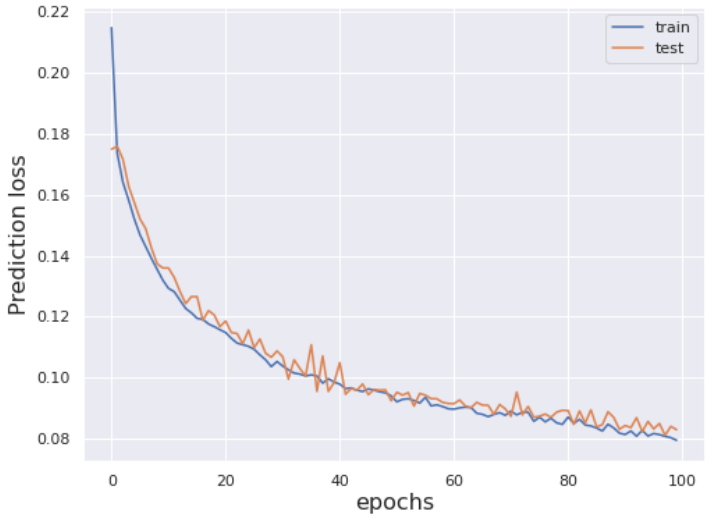
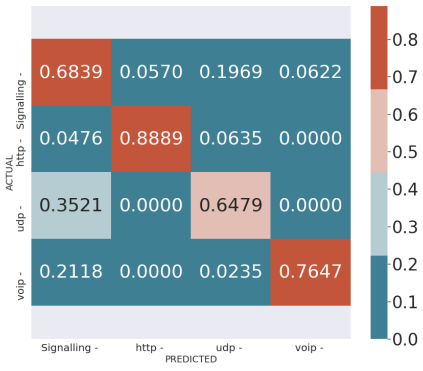
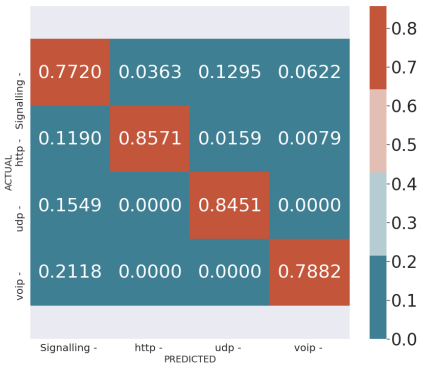


Figure 10: La performance des pertes de prédiction.



(a) Sans Auto-codeur



(b) Avec auto-codeur

Figure 11: Matrices de confusion dans la tâche de prédiction.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context

Mobile networks have impacted our social lives in many different ways, such as our professional networks, social interactions, communication, etc. There has been a huge progression in mobile wireless communications, from an initial expensive wireless technology to the popularity today. Until now, four cellular network generations, from 1G to 4G, have been deployed, and the fifth-generation (5G) has been introduced and applied in some countries. New use cases arise with each expansion of the cellular networks. The 1G and 2G were voice-centric systems. Later, the 3G and 4G also supported data packet services. Each later generation enhances the benefits of the previous generation and brings a significant technological evolution. Unlike the first four generations that provide more services to human users, 5G enables innovative solutions for both machines and humans.

In recent years, the Internet of Things (IoT) has been a hot issue, both in the research fields and on the industrial market. The expectation of IoT that everything such as devices, machines, and vehicles is going to be connected to the Internet is leading the trend. Cisco estimates that, by 2023, there will be 15 billion communicating devices in the world [1]. These devices can vary from indoor to outdoor, from smart meter's several bytes per day to vehicle monitor's huge amount of information per milliseconds, from Bluetooth, WiFi to sub-GHz technologies, and from battery-based such as wearable devices to always charged such as surveillance camera. With different features, their particular requirements vary.

In IoT networks, most of the devices do not have high demands in terms of latency and data rate. They only have a small amount of data to transfer and do not require a response with a milliseconds delay. Nowadays, these devices are usually connected through 802.11; however, such communications are limited in range. From a capacity point of view, an eNB can serve a high density of devices with tens of kilometers coverage. However, due to avoiding the frequent battery maintenance of the high number of devices, low energy consumption is a strong requirement for IoT. Therefore, the requirements for IoT networks are to have wide coverage, to be able to handle massive devices and to keep low energy consumption as well as low device cost.

In order to meet the requirement for the IoT world, recent technologies have come to the scene dedicated to Low Power Wide Area Network (LPWAN), such as NB-IoT, which has been included in the 3GPP standardization, and is the narrowband system with low-cost, low-power consumption based on LTE; LoRa, provides the possibility for people to build their own

LoRaWAN; and UNB (Ultra Narrow Band), developed and patented by the French company Sigfox, was the first one to be commercially initiated for IoT purpose.

In this thesis, we focus on 3GPP NB-IoT. This solution is expected to broaden the scope of 5G applications to include smart cities, object tracking, and remote control among others. NB-IoT can operate in a spectrum as narrow as 200 kHz by refarming GSM, LTE carriers, and reusing network infrastructures, which represents a major benefit for operators while being able to operate in the LTE spectrum. The main design objectives of NB-IoT are increased indoor coverage, support of a massive number of connections (up to 50000 per NB-IoT network cell), long battery life (up to 10 years), and low user equipment device complexity. In fact, NB-IoT is very flexible and can operate in any of the 2G/3G/4G band (from 450 Mhz to 3-5 GHz) since it achieves an excellent co-existence and compatibility performance with legacy cellular systems. NB-IoT provides more reliability and more quality of service (QoS) since it operates in a small portion of the existing available cellular spectrum. In addition, NB-IoT can use the existing cellular network infrastructure, which reduces deployment costs.

1.2 Motivations

As stated previously, NB-IoT is a technology that enables long-range communications by using either GSM or LTE carriers, depending on the mode of operations. It allows long-range communications and a massive number of connected devices in an area, whereas IEEE 802.11 would have too much interference that would render it unusable.

Narrowband IoT can offer the best technologies in smart cities and agricultural projects. For example, in smart cities, NB-IoT technology can monitor air pollution, traffic, smart metering, and processes that require small data packets to be sent infrequently to the network center or to an IoT management platform. Since these devices can be located deep indoors, or even underground, this reduces coverage due to the attenuation introduced by the buildings. Furthermore, in dense urban areas, many of these devices usually use the same cell, causing the problem in Radio Resource Management.

As such, proper network dimensioning, taking into account the number of devices, their density, and their locations, is important to achieve the best performance. Otherwise, we might face an unable network to provide the services or even disrupting the services already implemented by overloading radio resources in a cell, which would render the communications unable to operate. In addition, in recent modern networks with limited and highly expensive time-frequency radio resources, classification and proactive knowing traffic arrival can significantly improve resource utilization. It allows the system for optimal resource allocation. Within this context, the main concerns when dimensioning NB-IoT networks are coverage (i.e., the number of devices connected per area unit), resource management (i.e, the number of required resource allocations to guarantee the network quality of service), and traffic pattern recognition and prediction (for traffic distribution prediction and predicting the future behaviour of users). Toward this end, we aim to use stochastic geometry and machine learning approaches to study the NB-IoT network dimensioning.

1.2.1 Stochastic Geometry Tool for Analyzing NB-IoT

Although the initial idea of modeling cellular networks by using stochastic geometry tools was started as early as 1997 [5], the key metrics such as coverage and rate were not in the scope of the study. Instead of assuming the location of base stations (BSs) is deterministic as in

regular grid models, the stochastic geometry model assumes the BSs are randomly placed in the space following a certain point process. So far, there are various point processes that have been proposed to model cellular networks, such as homogeneous PPP [6], determinantal Point Processes [7], hard-core processes [8], and so on. Among all these point processes, homogeneous PPP is the most widely used model due to its tractability and accuracy. Therefore, this thesis focuses on homogeneous PPP as well as its characteristics.

In the PPP model, the locations of BSs (or UEs) are modeled as a realization of PPP with intensity λ. The PPP model is significantly more tractable than the traditional regular grid model and can evaluate a real deployment at least as accurately as the grid model. As the deployment of modern networks tends to be opportunistic, irregular, and dense, the PPP model is envisaged to be increasingly close to the real deployment.

The randomness of the locations of UEs and BSs actually makes the model more tractable and accurate to evaluate the key metrics of system performance such as coverage probability (distribution of signal-to-interference-plus-noise ratio (SINR)) and average achievable data rate.

In much literature, the distribution of SINR of a cellular network modeled by the Poisson process has been presented in closed-form expressions. However, there is a lack of research on applying stochastic geometry to derive the performance of a system in NB-IoT networks. In the next chapter, the NB-IoT network studied in this thesis is introduced along with the contributions based on the stochastic geometry approach.

1.2.2 Machine Learning approach

Starting from Long Term Evolution (LTE), the concept of Self-Organizing Networks (SONs) has been introduced by 3GPP Release 8 [9]. It intends to decrease network deployment times, reduce congestion, improve throughput and reduce installation and management costs. However, several methods provided for SONs, such as the Markov model or genetic algorithms, are both inefficient and costly for mobile operators. Most of these solutions require manual data analysis and adjustment of the system parameters to optimize the network. Besides, through the functionalities of self-configuration, self-optimization, and self-healing, SONs focus on increasing network automation and thus minimize human intervention in cellular network management. This objective can be done with a Machine Learning approach.

In recent years, Machine Learning (ML) is considered one of the most potential and promising tools to investigate a large set of complex problems. ML techniques, in particular Deep Learning (DL) algorithms, have obtained tremendous impacts in many research fields, becoming in a relatively short amount of time the state-of-the-art framework for a wide range of problems. The rapid growth of ML is directly correlated with the technology evolution: training neural networks has become relatively fast and cheaper, thanks to more powerful hardware, like Graphical Processing Units (GPUs) and thanks to specifically designed Tensor Processing Units (TPUs) for Deep Learning [11]. Moreover, the rapid growth of the AI community has been beneficial to the development of many open-source libraries and tools (e.g., TensorFlow, Keras, PyTorch), that helps many researchers in the implementation and deployment of ML algorithms.

However, there is a significant limitation for the advancement of Machine learning-based research in wireless and mobile networks due to the lack of extensive datasets. In fact, retrieving data from mobile networks is more difficult and Mobile Network Operators (MNOs) are not always favorable to release data due to security and user privacy issues.

The lack of extensive datasets represents a major limitation for the advancement of ML-based network management research. To assist resource allocation, we need to have finer quality data, accessing directly to the network channel information exchanged between the users and the associated base stations. This allows having access not only to aggregate base station statistics but also to more information derived from the radio protocols, such as the resource block allocation and the link adaptation mechanism of the system. The traffic data used in this thesis comes from the Amarisoft LTE Web GUI, which allows to analyze LTE software logs and get real-time information from the system.

1.3 Research Contributions

The main contributions of this thesis are to study the dimensioning of the NB-IoT network using stochastic geometry and machine learning techniques. To that end, this thesis addresses the following questions:

1. *Focusing on the deployed NB-IoT networks, what are the coverage performances of single-cell and multi-cell NB-IoT networks?*

   To answer this question, we will model the NB-IoT network, where the UEs and eNBs are located according to a homogeneous Poisson point process. Then, based on the stochastic geometry approach, we derive an analytical expression of the coverage and successful transmission probability for a single-cell NB-IoT network. In the multi-cell scenario, we use the machine learning approach to predict the coverage performance based on the network parameters.

2. *How the network outage probability depends on the configuration network parameters, including the network load, the user density, and the required QoS?*

   To answer this question, we consider the radio resource management problem. We define the network outage as the event that occurs when the number of request RBs exceeds the number of available ones. We first derive the expression of the outage probability considering network parameters. Then, based on a given outage probability, we determine the number of required RBs in the NB-IoT network to guarantee the target outage probability, taking into account the repeated transmission feature in NB-IoT.

3. *How can be improved resource utilization in recent modern networks?*

   To answer this question, we utilize the mobile traffic information that is collected from the Downlink Control Information (DCI) using the Amarisoft testbed. Based on the collected data, we design a deep learning model for traffic pattern recognition and prediction of the type of arrival packet using Long Short-Term Memory (LSTM) neural networks.

1.4 Thesis Structure

The rest of this thesis is structured as follows:

Chapter 2. State of the art. The thesis introduces the state of the art of IoT, the emergence of LPWA, and the NB-IoT problematic introduction.

Chapter 3. Fundamental Concepts. This chapter gives an introduction to the Poisson point process and Machine learning that we use throughout this thesis. We first define Poisson point process in an understandable way and present some of its important properties, such as the distribution of the number of points and the Campbell theorem or some operations preserving the Poisson law. We also present some examples of Poisson point process applications. Then, we present an overview of Machine learning algorithms and the fundamental principles of Deep learning algorithms. The advantages and limitations of Deep learning in the wireless network are also derived at the end of this Chapter.

Chapter 4. Performance evaluation of coverage in NB-IoT networks. The chapter proposes an analytical model to study the coverage and the successful transmission probability in a single-cell NB-IoT network based on stochastic geometry. In the case of multi-cell, we use the machine learning approach to derive the model that can predict the coverage performance based on the input network parameters.

Chapter 5. NB-IoT network resource outage and dimensioning. This chapter focuses on the dimensioning in the downlink of a multi-cell NB-IoT network. We use stochastic geometry tools and the properties of Bell Polynomials to provide an analytical model to estimate the required number of resource blocks to avoid the radio resource outage.

Chapter 6. Traffic pattern recognition and prediction. This chapter proposes to use Deep learning approach to recognize and predict the cellular traffic patterns. More precisely, we use the information extracted from the Downlink Control Information (DCI) and apply the Long Short-Term Memory to capture the relation between the applications from the control channel information.

Chapter 7. Conclusions and perspectives. This chapter draws the main conclusion, and outlines the main contributions of this thesis and the perspectives.

1.5 Publications List

Here is the list of publications:

- **T. A. Nguyen**, P. Martins, V. T. Nguyen and T. M. T. Nguyen, "A New Analytical Model for the Performance Evaluation of the Uplink Transmission in NB-IoT Networks," *IEEE 88th Vehicular Technology Conference (VTC-Fall)*, pp. 1-5, Chicago, USA, Aug. 2018.

- **T. A. Nguyen** and P. Martins, "Multi-cell Downlink Dimensioning in NB-IoT Networks," *16th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 117-122, Thessaloniki, Greece, 2020.

- **T. A. Nguyen** and P. Martins, "Cellular Traffic Type Recognition and Prediction," *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2021.

Chapter 2

State of the art

In this chapter, we present an overview of the Internet of Things (IoT) concerning its evolution from wireless sensor networks, and its connectivity technologies. Among these enabling technologies, LPWA (Low Power Wide Area Network) stands out as an ideal candidate for the numerous power-critical, long-range, and low-throughput IoT devices. The characteristics and existing technologies of LPWA are presented. Then we focus on the Narrow-Band Internet of Things (NB-IoT) that I worked on during this thesis. Finally, we summarize and discuss some relevant and open issues in the research field.

2.1 IoT review

2.1.1 Evolution of IoT

As a big part of the evolution predicted in the fifth generation 5G, the Internet of Things (IoT), whose key idea is to connect everything and everyone by the Internet, has gained a lot of attention in recent years. Its emergence is necessary as the result of the advancements of wireless communication technology [10]. IoT is broadening at a speedy rate, which promoted several novel services in different application domains, including, but not limited to, Industry 4.0, Smart Cities, Intelligent Transportation Systems, Precision Agriculture, healthcare, and environmental monitoring.

In the beginning, the services offered by IoT were focused on Smart home that is realized by machine to machine (M2M) systems. They connect devices at home such as thermometers, energy meters, lighting control systems, music streaming and control systems, etc. Most of these systems have some connectivity through an application so that a user can manage them through a standard Web browser or a smartphone. But the scale of devices was very limited; most M2M communication solutions are purpose-built and designed to satisfy a very particular application and communication needs. Gradually, the application in M2M has been enlarged to all objects and even living things, such as health surveillance, smart grids, and smart cities. Thus not only the number of people learning the advantages of IoT is growing, but also the number of objects to be connected is rising. Currently, there are more than 28 billion connected devices worldwide. Cisco shows that the number of networked devices per capita is 3.4 in 2020. By 2024, the overall industry is expected to generate a revenue of $4.3 trillion, coming from different sectors, such as device connectivity, manufacturing, and other value-added services [11]. Machine-to-machine (M2M) communications or Machine-type

communications (MTC) play an essential role at the core of IoT for the connectivity between devices and the cloud [12].

Traditionally, the key drivers in existing cellular network technologies such as Global System for Mobile (GSM) or Long Term Evolution (LTE) were designed for human-centric communications and are not a perfect fit for IoT applications. IoT applications do not need all capabilities of these access technologies; moreover, the associated costs would be prohibitive. They have different characteristics in terms of traffic pattern (e.g., periodic, low data rate), latency requirement (e.g., delay sensitivity), or deployment density [13]. The main considerations of those use cases for the underlying radio technologies are low power consumption, low data rate, scalability, and big coverage [14]. Fundamental design changes are needed from end devices to the network to fulfill these challenges. 5G system design must consider these requirements and guidelines in developing prospective future technologies.

2.1.2 Enabling technologies in IoT

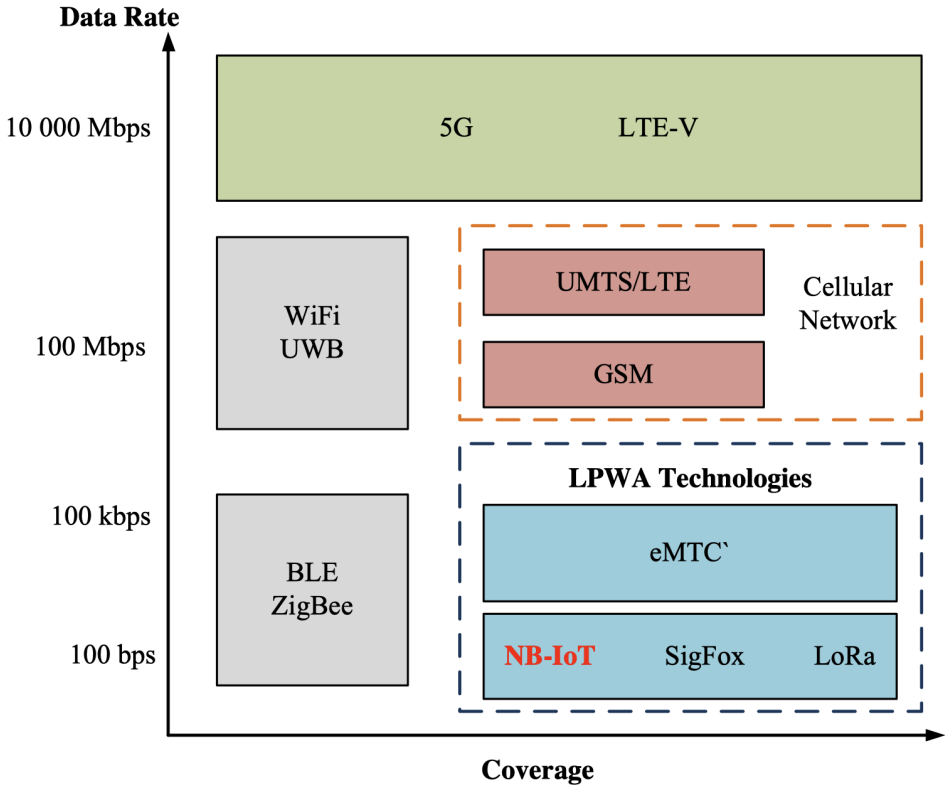


Figure 2.1: Wireless communication technologies.

The IoT technologies can be separated from the aspect of communication distance: short-range communication technologies (such as Zigbee, Wi-Fi, Bluetooth, etc.) and Low Power Wide Area (LPWA) technologies based on cellular networks, e.g., Global System for Mobile Communications (GSM), Long-Term Evolution (LTE), etc. The comparison between several communication technologies from different perspectives is present in Figure 2.1. The part

below will introduce the representative technologies of each type.

- **High data transmission rate.** The available access technologies are Wi-Fi and traditional cellular networks such as GSM, LTE. The data transmission rate is higher than 10 Mbps. While GSM and LTE provide long-range communication, Wi-Fi only allows devices to connect and exchange information within a short distance of 100m. They are mainly used in direct television transmission, vehicle navigation systems, smart grids, etc. Their good features such as high bandwidth, high data rate, low latency certainly cause high energy consumption, so they do not fit applications that are power-critical.
- **Medium data transmission rate.** The data transmission rate is lower than 1 Mbps. The available access technologies are 2G and MTC/eMTC. Such applications include POS machines, smart homes and etc. The specific communication technology is in use, like Near Field Communication (NFC). NFC enables two electronic devices to communicate within 10 cm and provides a data rate ranging from 106 to 424 kbps. It is mostly used in smart mobile payments, credit card contactless payments, and electronic tickets.
- **Low data transmission rate.** The data transmission rate is lower than 100 Kbps. The available access technologies are NB-IoT, SigFox, LoRa, and short-range wireless communications like Zigbee. In short-range and low-data transmission communication technologies, the maximal coverage area is 100m, and the data transmission rate can be up to 100 Kbps. On the other hand, the long-range and low-data transmission communication technologies are mainly applied in LPWA networks, including sensors, smart metering, logistics, and intelligent agriculture. LPWA is a novel paradigm that treats the trade-off between communication range and energy efficiency. It is promising to achieve a few to tens of kilometers of connectivity range and a battery life of ten years.

To give an more global insight into the LPWA network, each of the technologies will be introduced in more detail in the section below.

2.2 The emergence of LPWA

Low Power Wide Area (LPWA) networks represent a novel communication paradigm, which will complement traditional cellular and short-range wireless technologies in addressing diverse requirements of IoT applications. LPWA technologies offer unique sets of features, including wide-area connectivity for low power and low data rate devices, which are not provided by legacy wireless technologies.

LPWA networks are unique because they make different tradeoffs than the traditional technologies including short-range wireless networks and cellular networks. First, the range of non-cellular wireless technologies is limited to a few hundred meters. Therefore, they can not provide communication over large geographical areas. The devices cannot be deployed arbitrarily, a requirement for many applications for smart cities, logistics and personal health. Second, while traditional cellular networks provide wide area coverage but they do not achieve energy efficiency high enough to offer ten years of battery lifetime.

With a communication range of a few to tens of kilometers and battery life of ten years and beyond, LPWA technologies are the solution for low-power, low-cost and wide-range communications. Although LPWA technologies are not suitable for many industrial IoT, vehicle

to vehicle (V2V), and vehicle to infrastructure (V2I) applications, they still meet various applications for smart cities, smart metering, wearable devices, environmental monitoring, etc. that require a small amount of data exchange and also infrequently. This is the reason why LPWA technologies still have much attention after the appearance of proprietary technologies such as SigFox or LoRa.

2.2.1 Characteristics of the LPWA

As the IoT market rapidly expanded, LPWA is expected to provide ubiquitous connectivity for smart cities or rural areas. It moves from the data rate expansion, to the upcoming number of devices growth in the dense wireless sensor networks. The main characteristics of the LPWA are to achieve a long-range with low power consumption and low cost, unlike that of the other technologies for which achieving higher data rate, lower latency, and higher reliability.

- **Long Range.** LPWA technologies are designed for wide-area coverage and hard-reach indoor places such as basements. The target is a 20 dB gain over legacy cellular systems. To achieve this goal, Sub-GHz band and special modulation schemes such as narrow-band modulation and spread spectrum techniques have been applied [15].

- **Low Power Consumption.** LPWA networks are designed to prolong battery lifetimes by placing devices in low-energy sleep mode and only waking them when they need to communicate. In addition, data has to be sent with the least emission power, and with the smallest time duration. Therefore, the device is possible to operate for several years.

- **Low Cost.** The user devices are awake only when they have data to send and have to be simple, light, and with low computation capabilities. Therefore, the cost of such a device can be as low as ten dollars or even less. For example, LoRa wide area network and SigFox devices cost approximately 2-5$ each.

  The use of star-type (instead of mesh) connectivity, simple MAC protocols, and techniques to offload complexity from end devices enables manufacturers to design simple and, therefore, low-cost and devices.

2.2.2 Existing technologies of LPWA

There are currently several LPWA technologies, each employing different techniques to meet the Machine-to-Machine (M2M) requirements. In the unlicensed spectrum, LoRa and SigFox are the most common. However, the licensed spectrum is widely known to supply higher reliability and Quality of Service. Among the licensed LPWA networks, Narrow-band IoT (NB-IoT) has been recognized as a promising and effective technology. We highlight some key points of these technologies below:

- **LoRa.** LoRa stands for Long-Range, a physical technology for long-range, low-power wireless communication systems invented by Semtech. Among other LPWA technologies, LoRa is one of the most promising and widely adopted technologies. It has the robustness to interference and long-range coverage. To mitigate interference, end devices select channels in a pseudo-random fashion for every transmission. To provide long-range communication, LoRa utilizes a spread-spectrum technique with Frequency Shifting Keying (FSK) modulation that can even demodulate the data below the noise floor [16]. LoRa can support data rates from 0.3 kbps to 50 kbps.

- **SigFox.** SigFox is another proprietary LPWA solution that operates in the 200 kHz band of the publicly available spectrum at a data rate of 100 bps or 600 bps. By using ultra narrowband (UNB), SigFox achieves a range between 10 and 50 km and experiences very low noise levels, resulting in high receiver sensitivity, ultra-low power consumption. SigFox relies on the cloud for computing and pushes the network complexity to the base station to achieve affordable and energy-efficient devices.

- **NB-IoT.** NB-IoT is one of LPWA solutions designed for IoT applications, which uses mobile network providers, was designed and standardized by 3GPP. NB-IoT aims at enabling deployment flexibility, long battery lifetime, low device cost and complexity, and coverage extension. The data rate is limited to 250 kbps for the multi-tone downlink communication and 20 kbps for the single-tone uplink communication.

2.3 Narrow Band IoT

NB-IoT is a new 3GPP radio access technology that is designed to archive excellent coexistence performance with legacy GSM and LTE technologies. NB-IoT requires 180 kHz minimum system bandwidth for both downlink and uplink. The choice of system bandwidth enables a number of deployment scenarios. A GSM operator can replace one GSM carrier (200 kHz) with NB-IoT. In contrast, an LTE operator can deploy NB-IoT inside an LTE carrier by allocating one of the physical resouce blocks (PRBs) of 180 kHz to NB-IoT (in-band or guard-band deployments).

NB-IoT reuses the LTE design extensively, including the numerologies, downlink orthogonal frequency-division multiple access (OFDMA), uplink single-carrier frequency-division multiple access (SC-FDMA), channel coding, interleaving, and so on. This significantly reduces the time required to develop full specifications. In the following parts, we provide a state-of-the-art overview of the air interface of NB-IoT with a focus on the key aspects where NB-IoT deviates from LTE.

2.3.1 Architecture and deployment scenarios

Architecture

NB-IoT uses the same network architecture as in the LTE network but with some optimizations to meet the requirements of NB-IoT devices. NB-IoT architecture is based on the Evolved Packet System (EPS), as shown in Figure 2.2. A new node has been added to the architecture, known as Service Capability Exposure Function (SCEF), designed for machine-type data. Two optimizations are defined for Cellular IoT (CIoT) in Evolved Packet System (EPS): *Control plane CIoT EPS optimization* (marked as red lines), and *User plane CIoT EPS optimization* (blue lines). Both optimizations may be used for sending data to the correspondent application.

With the User Plane CIoT EPS optimization, IP and non-IP data are transferred in the same way as conventional data traffic, i.e., over radio bearers via the Serving Gateway (SGW) and the Packet Data Network Gateway (PGW) to reach the application server. On the Control Plane CIoT EPS optimization, UL data are transferred from the eNB (CIoT RAN) to the Mobile Management Entity (MME). From there, they may either be transferred via Serving Gateway (SGW) to the Packet Data Network Gateway (PGW) or to the Service

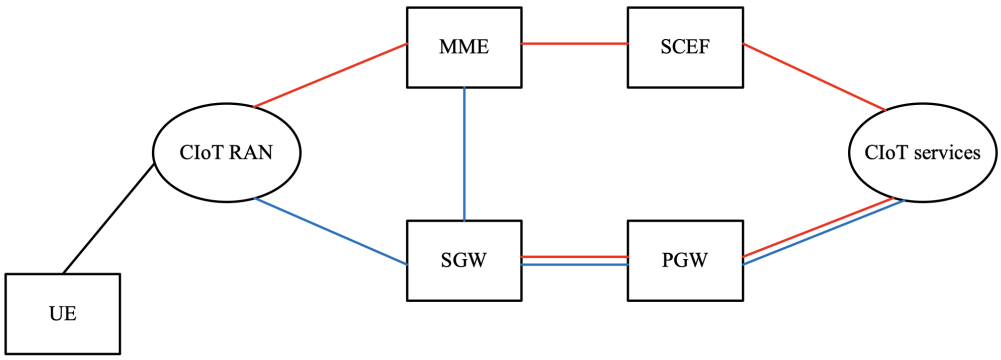


Figure 2.2: NB-IoT architecture.

Capability Exposure Function (SCEF), which is a new node responsible only for non-IP data. Finally, the UL data are forwarded to the application server (CIoT services). DL data are transmitted over the same ways but in the reverse direction.

Deployment scenarios

NB-IoT technology occupies a frequency band of 180 kHz bandwidth, which corresponds to one resource block in LTE transmission. There are three different deployment scenarios, such as standalone, in-band, or guard-band, as depicted in Figure 2.3.

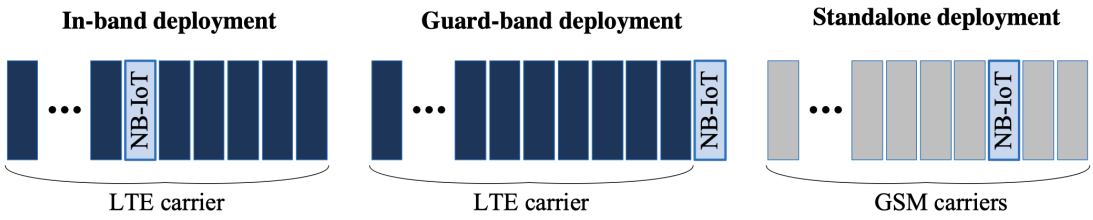


Figure 2.3: Three scenarios of NB-IoT deployments.

- *Standalone deployment.* NB-IoT occupies one GSM carrier of 200 kHz. There is still a guard interval of 10 kHz remaining on both sides of the spectrum. This deployment mode also works without neighboring GSM carriers.
- *Guard-band deployment.* Utilizes the unused resource blocks within an LTE carrier's guard-band.
- *In-band deployment.* Utilizes the resource blocks of 180 kHz within an LTE carrier. Although the assignment of resources between LTE and NB-IoT is not fixed, not all resource blocks within the LTE carrier are allowed to be used for cell connection.

The deployment scenario should be transparent to a user equipment (UE) when it is first turned on and searches for an NB-IoT carrier. Similar to existing LTE UEs, an NB-IoT UE is

only required to search for a carrier on a 100 kHz raster. An NB-IoT carrier that is intended for facilitating UE initial synchronization is referred to as an anchor carrier. The 100 kHz UE search raster implies that for in-band deployments, an anchor carrier can only be placed in certain PRBs.

In order to cope with different radio conditions, NB-IoT introduces three kinds of coverage enhancement (CE) levels, including normal coverage, robust coverage, and extreme coverage, which correspond to the minimum coupling loss (MCL) of 144 dB, 158 dB, and 164 dB respectively, as shown in Figure 2.4. The main impact of the different CE levels is that the messages have to be repeated several times.

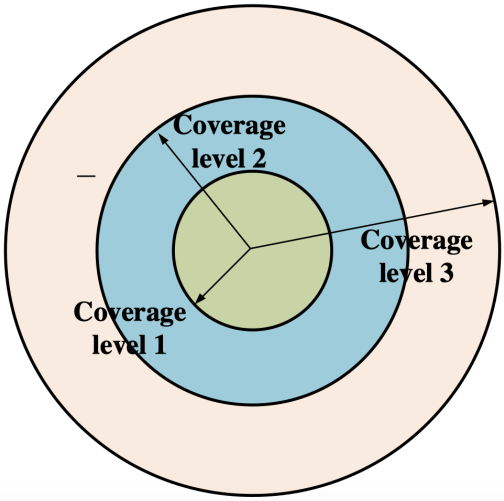


Figure 2.4: Three scenarios of NB-IoT deployments.

Downlink transmission scheme

The downlink transmission scheme of NB-IoT is based on OFDMA with the same 15 kHz subcarrier spacing as LTE. Slot, subframe, and frame durations are 0.5 ms, 1 ms, and 10 ms, respectively, identical to those in LTE. Basically, an NB-IoT carrier uses one LTE PRB in the frequency domain (i.e., 15 kHz subcarrier) for a total 180 kHz. Reusing the same OFDMA numerology as LTE ensures good coexistence performance with LTE in the downlink. For example, when NB-IoT is deployed with "in-band" scenario, the orthogonally between all the other LTE PRBs and the NB-IoT PRB is maintained in the downlink.

Uplink transmission scheme

The uplink of NB-IoT supports both multi-tone and single-tone transmission. Multi-tone transmission is based on single-carrier frequency-division multiple access (SC-FDMA) using the same 15 kHz subcarrier spacing and 0.5 ms slot as LTE. Single-tone transmission support two numerologies, 15 kHz and 3.75 kHz. While the 15 kHz numerology is identical to LTE and achieves the best coexistence performance with LTE in the uplink, the 3.75 kHz single-tone numerology uses 2 ms slot duration. The uplink NB-IoT carrier uses a total system bandwidth of 180 kHz like the downlink.

2.3.2 NB-IoT physical layer

Physical downlink channels

NB-IoT provides the following physical signals and channels in the downlink:

- Narrowband primary synchronization signal (NPSS);
- Narrowband secondary synchronization signal (NSSS);
- Narrowband reference signal (NRS);
- Narrowband physical broadcast channel (NPBCH);
- Narrowband physical downlink control channel (NPDCCH);
- Narrowband physical downlink shared channel (NPDSCH).

	Subframe number									
Even numbered frame	0	1	2	3	4	5	6	7	8	9
	NPBCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPSS	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NSSS
	Subframe number									
Odd numbered frame	0	1	2	3	4	5	6	7	8	9
	NPBCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPSS	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH	NPDCCH or NPDSCH

Figure 2.5: Time multiplexing of NB-IoT downlink physical channels and signals.

Unlike LTE, these NB-IoT physical channels and signals are primarily multiplexed in time. Figure 2.5 illustrates how the NB-IoT subframes are allocated to different physical channels and signals. Each NB-IoT subframe spans over one PRB in the frequency domain and 1 ms in the time domain.

NPSS and NSSS are used by an NB-IoT UE to perform cell search, which includes time and frequency synchronization, and cell identity detection, respectively. In the case of in-band and guard-band mode, NPSS/NSSS signals can only be transmitted in a certain subset of the available LTE PRB locations, shown in Figure 2.6. This is due to the frequency offset between the DC carrier and the center of the NB-IoT carrier that should be held within 7 kHz range to ensure efficient cell searching. NPSS is transmitted every 10 ms and NSSS every 20 ms.

NPBCH carries the Master Information Block - Narrowband (MIB-NB) over an 80 ms block. This transmission is repeated 8 times where MIB remains unchanged over the 640 ms Transmission Time Interval (TTI) to cope with extreme coverage conditions. Figure 2.7 shows the NPBCH transmission and the location of the NRS signals. An NRS is used to provide phase reference for the demodulation of the downlink channels. NRSs are time-and-frequency multiplexed with information-bearing symbols in subframes carrying NPBCH, NPDCCH, and NPDSCH, using 8 resource elements per subframe per antenna port. NB-IoT supports up to two NRS ports. MIB-NB is a 50-bits size block that contains 16-bit CRC and spare bits. This block is used to provide an NB-IoT End Device (ED) with the main information like System

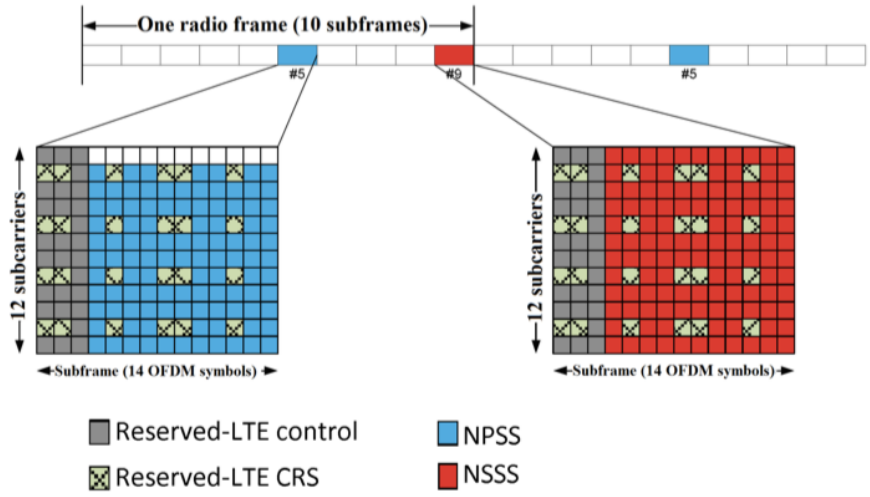


Figure 2.6: NPSS and NSSS Transmission [2].

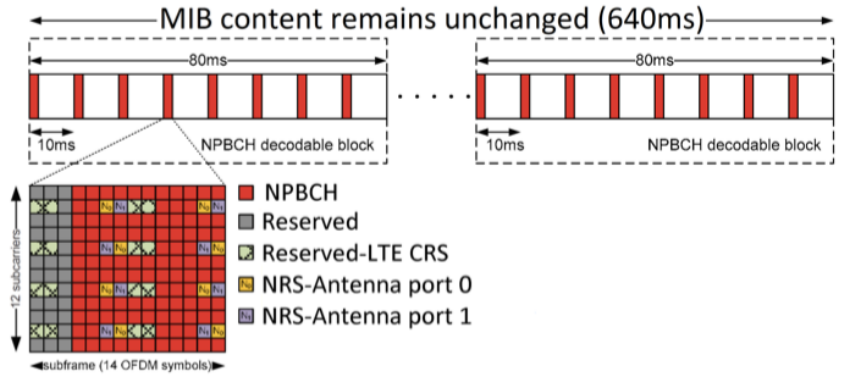


Figure 2.7: NPBCH Transmission [2].

Frame Number (SFN), the operational mode, LTE Cell-specific Reference Signal (CRS), and System Information Block (SIB) scheduling.

NPDCCH carries scheduling information for both downlink and uplink data channels. It further carries the HARQ acknowledgment information for the uplink data channel and paging indication and random access response (RAR) scheduling information. Data may be carried by one or two subsequent Narrow Band Control Channel Elements (NCCEs) during a sub-frame. As shown in Figure 2.8, each NCCF consists of six sub-carriers in a sub-frame.

Repetition of transmission is used in NB-IoT to achieve coverage enhancement. Each ED is configured to transmit NPDCCH several times based on the R_{max} which is chosen from $1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048$, depends on its coverage level. The number of repeated transmissions is also indicated in the DCI. An ED can determine the end of the NPDCCH transmission when it successfully decodes the NPDCCH before the last repetition.

NPDSCH carries data from the higher layers as well as paging messages, system information, and RAR message. The maximum transport block size (TBS) that can be transmitted

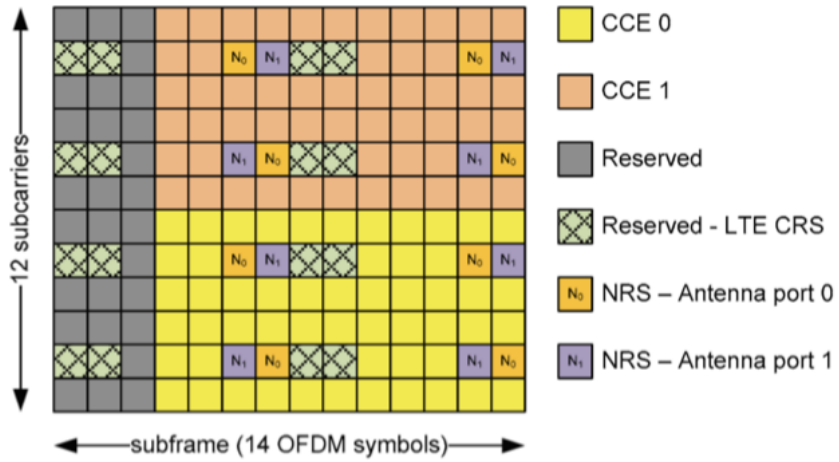


Figure 2.8: CCE Allocation in NPDCCH (in-band deployment) [2].

on the NPDSCH is limited to 680 bits.

Physical uplink channels

NB-IoT includes the following channels in the uplink:

- Narrowband physical random access channel (NPRACH);
- Narrowband physical uplink shared channel (NPUSCH).

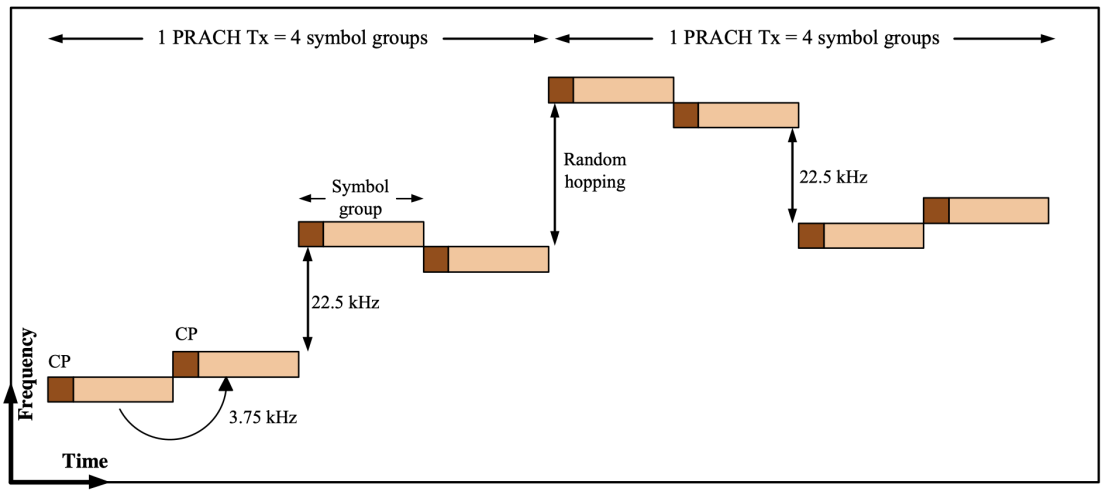


Figure 2.9: An illustration of NPRACH frequency hopping [2].

Since the LTE physical random access channel (PRACH) uses a bandwidth of 1.08 MHz, more than NB-IoT uplink bandwidth, so it is necessary a new NPRACH for NB-IoT. ED can

Subcarrier spacing (kHz)	Number of Tones	Number of SC-FDMA symbols	Transmission time interval (ms)
15	1	112	8
	3	56	4
	6	28	2
	12	14	1
3.75	1	112	32

Table 2.1: NPUSCH Resource Unit Definition

use this signalling channel in the random access channel procedure for cell accessing, where the preamble is transmitted. A preamble is based on a single sub-carrier of a single group, with frequency hopping for a single user, as shown in Figure 2.9. One NPRACH preamble consists of four symbol groups, with each symbol group comprising one cyclic prefix (CP) and five symbols. Hopping is between group of symbols, whereas pseudo-random hopping concerns repetitions of groups. Different cell sizes can be achieved when using sub-carrier spacing of 3.75 kHz (i.e., a symbol length of 267 us) and two-cycle prefix lengths: 66.7 μs (10 km) and 267 μs (35 km).

NPUSCH is designed to carry uplink data and send HARQ ACK/NACK. It provides extended coverage, long battery life, and massive capacity. This channel has two formats: Format 1 and Format 2. Format 1 is used for carrying Uplink data and uses the same LTE error correction code. The maximum transport block size of NPUSCH Format 1 is 1000 bits, which is much lower than that in LTE. NPUSH Format 1 supports multi-tone transmission, as shown in Table 2.1.

In this case, the UE can be allocated with 12, 6, or 3 tones. While only the 12-tone format is supported by legacy LTE UEs, the 6-tone and 3-tone formats are introduced for NB-IoT UEs who cannot benefit from higher UE bandwidth allocation due to coverage limitation. Moreover, NPUSCH supports single-tone transmission based on either 15 or 3.75 kHz. To support efficient power amplifier operation, single-tone transmission uses the low peak-to-average power ratio (PAPR) modulation schemes such as $\pi/2$-binary phase-shift keying (BPSK) or $\pi/4$-quadrature phase-shift keying (QPSK). Format 2 is used for signaling HARQ acknowledgment for downlink channel NPDSCH, and uses a repetition code for error correction.

NPUSH Format 1 and Format 2 have 7 OFDM symbols/slot, but only one symbol in Format 1 is the demodulation reference symbol (DMRS), while there are three symbols as DMRS in Format 2. DMRSs are used for channel estimation.

2.3.3 Connection control

Random access procedure occurs after the cell search has operated the downlink synchronization and performed before user equipment (UE) begins to communicate in the network. The UE sends an access request to the eNB, and then the eNB responds to the request and allocates a Random Access Channel (RACH). The UE obtains uplink synchronization signals from the network through random access and requests dedicated resources for data transmission.

Figure 2.10 illustrates UE states in NB-IoT. Each NB-IoT device has two states: RRC IDLE (sleeping state) and RRC CONNECTED (connections state). In NB-IoT, a UE connects

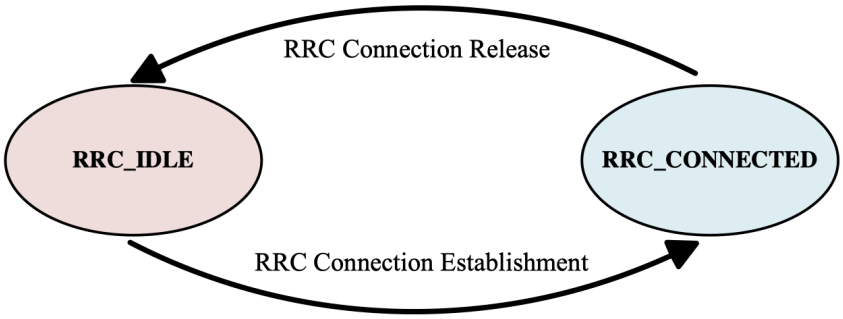


Figure 2.10: UE states in NB-IoT.

to only one eNB to communicate with. During movement, this UE may change its location several times, and when the connection is lost, it will search for a suitable eNB to connect. When UE has data to transmit, it will search for a cell on an appropriate frequency, read SIB information, and start the random access procedure. NB-IoT UE initiates a random access procedure when it meets one of the following conditions:

- The UE initiates a random access procedure on RRC IDLE mode or after being paged.
- The eNB needs to send downlink data to a UE in RRC CONNECTED mode and finds that the UE is out of synchronization in the uplink.
- The UE in RRC CONNECTED mode has data transmission requirements in the uplink but does not receive an Uplink Grant message from the eNB.

2.3.4 Random Access Procedure

Figure 2.11 shows the contention-based Random Access (RA) procedure, which consists of four steps.

Msg1. Random Access Preamble

Before transmitting random access preamble, an NB-IoT UE needs to confirm NPRACH resources that use the different coverage levels for NB-IoT coverage requirements. The system broadcast three sets of NPRACH configuration parameters corresponding to three coverage levels in NB-IoT. The UE sends a random access request in the selected coverage level based on its measured Reference Signal Received Power (RSRP) and the RSRP threshold configured for that enhanced coverage class. Then, the UE starts the RAT window, W. In addition, the way to generate a random access preamble of an NB-IoT UE is also different from LTE. The eNB no longer broadcasts the random access preamble number divided into group A and group B. Moreover, UE no longer needs to generate a random access preamble based on the preamble index. All UEs use a full 1 sequence as the default configuration.

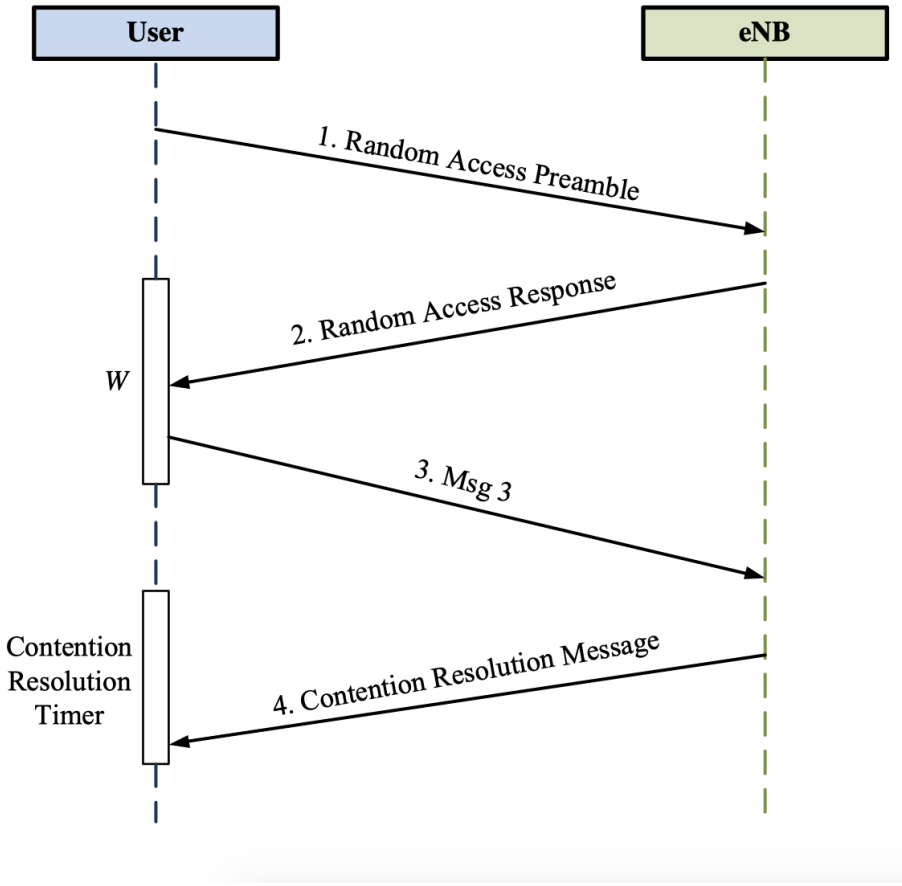


Figure 2.11: NB-IoT Random Access Procedure.

Msg2. Random Access Response

Upon preamble reception, the eNB applies for a Cell Radio Network Temporary ID (C-RNTI) and applies for scheduling uplink and downlink resources. The non-collided preambles can be detected by eNB with a detection probability. Then, the eNB sends a random access response (RAR) over the Downlink Shared Channel (PDSCH) for each UE. The response contains the RA preamble identifier, timing alignment information, initial uplink grant, and temporary C-RNTI. One PDSCH can carry random access responses to multiple UEs.

After sending the preamble, the UE monitors the NPDCCH and wait for a random access response within a random access response window:

- If the UE receives a response containing an RA-preamble identifier which is the same as the identifier contained in the transmitted random access preamble, the response is successful, and the RA-preamble identifier is used for uplink transmission;
- If the UE does not receive a response within the random access window W or fails to verify the response, the response fails. In this case, if the number of random access attempts is not reached the upper limit, the UE retries random access. Otherwise, random access fails.

The times of UE transmit preamble are counted at each coverage level. The counter adds one when UE fails to transmit a preamble at the current coverage level. If the counter reaches the maximum value, UE switches to the next higher coverage level, if exists. This behavior is repeated until the maximum number of transmissions that can be tried globally at all levels is reached. The original total counter is used as a total counter to determine whether the entire random access procedure fails or not.

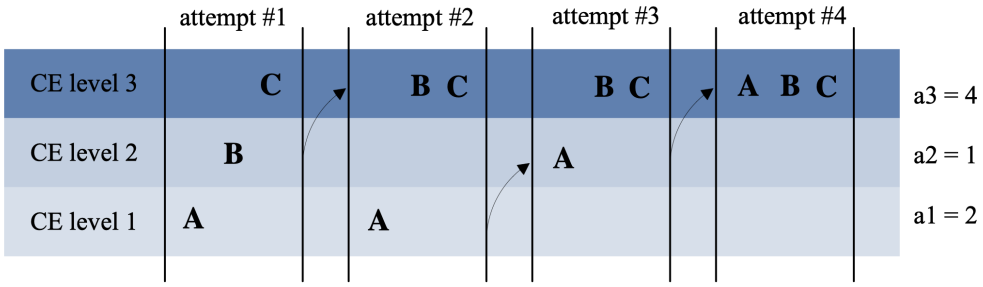


Figure 2.12: Coverage class hopping during random access procedure.

Figure 2.12 an example of this procedure. There are three different UEs, i.e., A, B, C, starting the first transmission attempt in the $CE1$, $CE2$, and $CE3$ level, respectively, based on the assumption that they always collide with other UEs. Note that the maximum number of preamble transmissions that a single UE can attempt in a general coverage level i is a_i and the maximum number of transmission is 4. In this case, if UE A fails in the second attempt, i.e., $a_1 = 2$, and it has to retry the subsequent transmission in the $CE2$ coverage level. Both A and B can try up to $a_2 = 1$ times in $CE2$ before jumping to the $CE3$ class in the immediately following attempt. On the other hand, UE C, whose level is $CE3$, is forced to persist in its level.

Msg3. Scheduled Transmission

The UE sends Msg3 using the dedicated resource. The UE transmits uplink scheduled data over the NPUSCH. The information in the transport block sent by the UE varies in different random access scenarios. A collision occurs when two or more devices transmit their messages through the same dedicated resource (due to selecting the same preamble in first step). Subsequently, the UE starts the Contention Resolution Timer.

Msg4. Contention Resolution

The eNB performs the contention resolution within the time length and sends it backs to the UEs through the Contention Resolution Identity (CRI) on the NPDSCH or the C-RNTI on the NPDCCH.

If the contention resolution is successful, the UE notifies upper layers, and stops the timer if the following conditions are met: the UE obtains the temporary C-RNTI from the NPDCCH, the MAC packet data unit (PDU) is successfully decoded, and the MAC PDU contains information matching the transmitted message in Msg3.

If the contention resolution message is not received, the UEs keep waiting for it up to the Contention Resolution Timer expiration. When the timer expires, the UE considers the

contention resolution as failed. The UE performs random access again if the number of random access attempts has not reached its upper limit. If the number of random access attempts has reached its upper limit, the random access procedure fails.

2.3.5 Repetition in NB-IoT

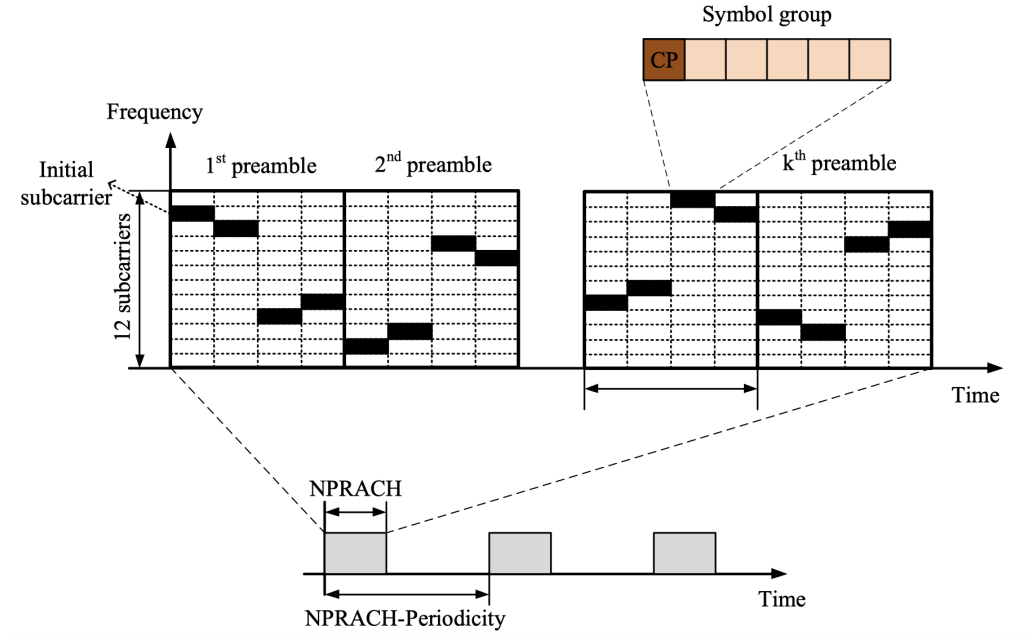


Figure 2.13: Structure of a preamble sequence.

In NB-IoT, the definition of *repetition* is introduced to provide coverage extension. An NB-IoT device transmits a preamble sequence through an NPRACH to start the RA procedure, as shown in Figure 2.13. The preamble sequence consists of k repeated preambles (where k is the repetition value, $k = 2^r$, $r = 0, 1, ..., 7$). The preamble sequence is differentiated based on the initial subcarrier carrying the first preamble. There are 12, 24, 36, and 48 orthogonal RA preamble sequences in each coverage enhancement (CE) level. During each NPRACH, the eNB determines the presence of the preamble sequence by comparing the accumulated received power of all symbol groups in a predefined detection threshold. The main difference between NB-IoT and LTE-A is that in LTE-A, we use random backoff and retransmission to increase the access success probability, while in NB-IoT, we use the repetition concept. In this way, in NB-IoT, we can reach a higher effective SNR even under correlated channel conditions.

2.4 Summary

In this chapter, we present an overview of IoT and its current technologies. We have sorted the enabling technologies into groups according to the data transmission rate. Among them, LPWAN has emerged as an effective way to cater to the demands of low-power and long-range transmission applications. The characteristics and existing technologies (i.e., LoRa, SigFox, NB-IoT) are also provided.

Particularly, we introduce NB-IoT with more details, as it builds the background of my thesis. We give the general background of NB-IoT and a brief review of architecture as well as deployment scenarios in NB-IoT networks. Then, we focus on aiming at the physical layer characteristics of NB-IoT. Lastly, we analyze the correlated uplink and downlink characteristics of NB-IoT and random access procedures in the NB-IoT network.

NB-IoT is an attractive new technology, enabling cost-efficient and flexible massive MTC deployments within cellular systems. The NB-IoT design has many similarities to LTE and can, in many cases, be deployed to an operator's existing LTE network. NB-IoT can operate in a narrow bandwidth and hence requires moderate fronthaul link capacity. Moreover, NB-IoT can benefit from deployment flexibility based on the loose latency requirements compared to LTE. NB-IoT has improved coverage compared to LTE, which comes from signal repetitions. In an ideally static channel, the uplink coverage gain can be achieved up to 20 dB from 128 repetitions.

The content presented in this chapter aims to provide a comprehensive overview and big-picture on the NB-IoT subject. There are still many open questions in NB-IoT networks. Some of them will be answered in the coming chapters.

Chapter 3

Fundamental Concepts

In this chapter, we review the main mathematical definitions used in this thesis. In Section 3.1, we first define the Poisson point process and study the applications Poisson point process. Then, we introduce some useful results applications concerning the Poisson point process in wireless networks. In Section 3.2, we present an overview of Machine Learning algorithms, which is expected to play a significant role in future wireless networks. Several deep learning architectures such as Multilayer Perceptron (MLP) and Recurrent Neural Network (RNN) are also presented in this Section. Finally, we give the advantages and limitations of applying deep learning techniques to wireless networks.

3.1 Background on Stochastic geometry

In this section, we explain why and how stochastic geometry in general and spatial Poisson point process, in particular, can help to model the wireless communications systems.

Nowadays, wireless communication systems consist of cellular networks, ad-hoc networks, sensor networks, Wi-fi networks, and others applied to all aspects of life. Contrary to wired systems, there is a much greater amount of uncertainly and randomness on node locations or channel conditions in the wireless system, and there is interference because many nodes share the same communication medium. A configuration of active users is never fixed in time: nodes can move or simply change their state. The received signal, which depends on the distance between the transmitter and the receiver, and the channel's condition as fading, can vary from time to time. Models for wireless communication systems must capture all these uncertainty and randomness. However, it is difficult to model a system perfectly: the more realistic a model is, the less tractable it is. There is always a balance between realistic modeling and tractability.

Why use Stochastic Geometry?

In general, we are interested in the SINR or SIR experienced at the network elements to analyze the performance of Evolved NodeB (eNBs) and User Equipments (UEs). In particular, the interference experienced at any network element is a function of the network geometry. It is influenced by not only the locations of interferers in the network but also their densities. Hence, the network geometry plays a significant role in the modeling of interference at receivers. For the last few years, under the influence of works of F. Baccelli, the models generating from stochastic geometry are gaining more and more attention. They enable us to represent the wireless network model more precisely and make calculations more rigorously. Stochastic

geometry is a powerful mathematical tool for modeling, analyzing, and designing wireless networks with random topologies. It is used to provide spatial averages of random patterns of points, taken over a large number of nodes at various locations or over many network realizations.

3.1.1 Point Process Essentials

In the general case, the point process $\Phi = \{X_i, i \in \mathbb{N}\}$ is a countable collection of randomly placed points residing in a measured place, which for cellular networks is the Euclidean space $\mathbb{R}^d$. A point process $\Phi \subset \mathbb{R}^d$ can be considered as a random closed set. There are several ways to describe a point process:

Random set formalism: A simple point process can be regarded as a countable random set $\Phi = \{x_1, x_2, ...\}$ with each element x_i being a random variable. If $N_1(A)$ and $N_2(A)$ are the numbers of nodes in A for two point processes Φ_1 and Φ_2, then $N(A) = N_1(A) + N_2(A)$ is the number of nodes in A of the superimposed process.

Random measure formalism: For a stochastic point process, we may define a point process by counting the number of points falling in any set $A \in \mathbb{R}^d$, i.e.

$$N(A) = \sum_{x_i \in \Phi} \mathbb{1}\left(x_i \in A\right) \tag{3.1}$$

Note that $N(A)$ is a random variable whose distribution depends upon Φ.

3.1.2 Poisson Point Process

Definition

Instead of assuming the location of eNBs is deterministic as in regular grid models, the stochastic geometry model assumes the eNBs are randomly placed in the space following a certain point process. The Poisson point process (PPP) is the most commonly used PP. The points in a PPP occur independently of one another. The PPP in $\mathbb{R}^d$ is such that for a compact set $A \subset \mathbb{R}^d$, the number of points in A has a Poisson distribution with mean $\lambda(A) = \int_A \lambda(x)dx$. In addition, if $A_1, A_2, ..., A_m$ are disjoint bounded sets, then the number of points in each of the disjoint sets is an independent random variable.

There are various point processes that have been studied in the cellular network, such as homogeneous PPP, determinantal Point process, hard-core processes, and so on. Among all these point processes, homogeneous PPP is the most widely used model due to its tractability and accuracy. In particular, a homogeneous PPP is a PPP with an intensity function that is constant over the space of interest, i.e., $\lambda(x) = \lambda$. In this case, the number of points in a compact set $A \subset \mathbb{R}^d$ simplifies to a Poisson random variable with mean $\Lambda(A) = \lambda \int_A dx = \lambda|A|$, where $|A|$ denotes the Lebesgue measure (i.e., size) of the set A. The Lebesgue measure is a standard way of computing the size of a subset of an d-dimensional Euclidean space in measure theory. In particular, for $d = 1$ it corresponds to the length of an interval, for $d = 2$ it is the area of the subset, and for $d = 3$ it is the volume of the subset.

Thus, the number of points in the compact set A, $N(A)$, has the following probability mass function (PMF):

$$\mathbb{P}\left(N(A) = k\right) = \frac{(\lambda|A|)^k \exp\left(-\lambda|A|\right)}{k!} \tag{3.2}$$

Example. *Consider a PPP in $\mathbb{R}^d$ with intensity λ having units of points/area. If A is a set denoting a circle of radius r, we would have $|A| = \pi r^2$ and $\Lambda(A) = \lambda\pi r^2$. The probability that there are k points in A is given by*

$$\mathbb{P}\left(N(A) = k\right) = \exp\left(-\lambda\pi r^2\right)\frac{(\lambda\pi r^2)^k}{k!} \tag{3.3}$$

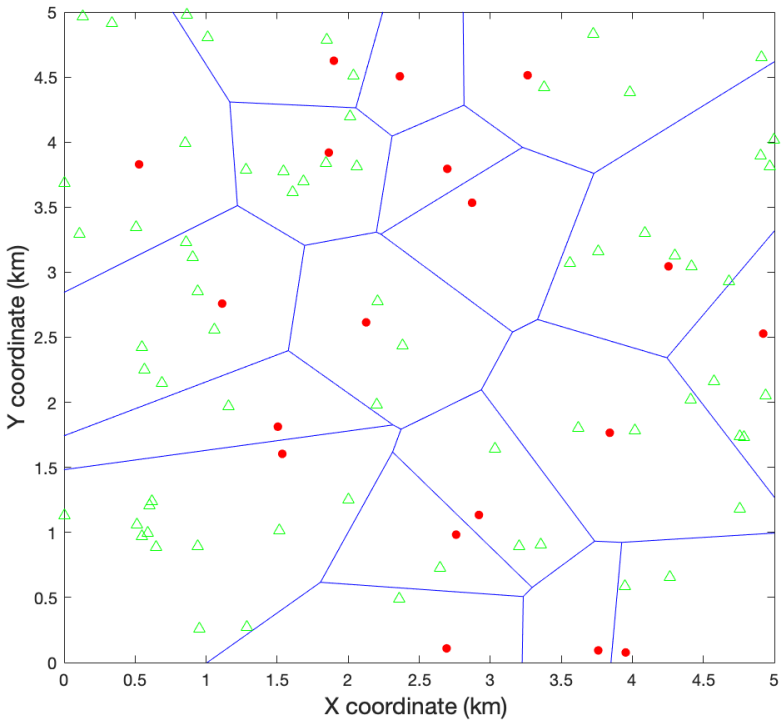


Figure 3.1: The deployment of a homogeneous network: the red points represent eNBs and green triangles represents UEs. The cell boundaries are shown and form a Voronoi Tessellation.

In the following, we describe some important theorem and properties for the homogeneous PPP.

Campbell's theorem:

Campbell's theorem can be used to compute the expectation of a random variable of the form $f(x)$ and thus provides a key tool to convert a sum into an integral. It states that if $\Phi \subset \mathbb{R}^d$ is stationary with intensity λ, the sum $\sum_{x\in\Phi} f(x)$ is a random variable with mean:

$$\mathbb{E}\left[\sum_{x\in\Phi} f(x)\right] = \lambda \int_{R^d} f(x)dx \tag{3.4}$$

Proof: We proof that the theorem holds for simple f, i.e., if f is a step function

$$f = \sum_{i=1}^{m} c_i \mathbb{1}_{A_i}$$

for compact $A_i \subset \mathbb{R}^d$ and $c_i \in \mathbb{R}$. We have

$$S = \sum_{x \in \Phi} f(x) = \sum_{x \in \Phi} \sum_{i=1}^{m} c_i \mathbb{1}_{A_i}(x) = \sum_{i=1}^{m} c_i \Phi(A_i)$$

and thus

$$\mathbb{E}(S) = \mathbb{E}\left(\sum_{i=1}^{m} c_i \Phi(A_i)\right) = \sum_{i=1}^{m} c_i \mathbb{E}\Phi(A_i) = \sum_{i=1}^{m} c_i \Lambda(A_i) = \int_{\mathbb{R}^d} f(x)\Lambda(dx)$$

Campbell's theorem can be used to compute the mean and variance interference in a cellular system.

Probability Generating Function (PGFL):

The PGFL is useful for converting an expectation of a product of the points in the PPP into an integral. The PGFL of a homogeneous Poisson point process is given as

$$\mathbb{E}\left[\prod_{x \in \Phi} f(x)\right] = \exp\left(-\lambda \int_{R^d} (1 - f(x))dx\right) \tag{3.5}$$

Proof: Following the characteristic functional, we set:

$$f(x) = e^{-v(x)} = 1 - \sum_{i=1}^{n} (1 - z_i)\mathbb{1}_{C_i}(x)$$

for $z_i \in [0, 1]$ and $C_1, C_2, ..., C_n$ pairwise disjoint and compact. Then

$$\begin{aligned} \mathbb{E}\left(\prod_{x \in \Phi} f(x)\right) &= \int_N z_1^{\varphi(C_1)} . \cdots . z_n^{\varphi(C_n)} P(d\varphi) \\ &= \mathbb{E}\left(z_1^{\Phi(C_1)} . \cdots . z_n^{\Phi(C_n)}\right) \\ &= \mathbb{E}\left(z_1^{\Phi(C_1)}\right) . \cdots . \mathbb{E}\left(z_n^{\Phi(C_n)}\right) \end{aligned} \tag{3.6}$$

By applying the moment-generating function for the Poisson distribution, we have $\mathbb{E}\left(z_1^{\Phi(C_1)}\right) = \exp(-\Lambda(C_1)(1 - z_1))$. Then

$$\mathbb{E}\left(\prod_{x\in\Phi} f(x)\right) = \exp\left(-\Lambda(C_1)(1-z_1)\right).\cdots.\exp\left(-\Lambda(C_n)(1-z_n)\right) \tag{3.7}$$
$$= \exp\left(-\sum_{i=1}^{n}\int_{C_i}(1-z_i)\Lambda(dx)\right)$$
$$= \exp\left(-\int_{\mathbb{R}^d}(1-f(x))\Lambda(dx)\right)$$

A PGFL completely characterizes a simple point process and can be used to derive all the density measures of point process. An important application of PGFL is the Laplace transform of a random variable.

$$L_\Phi(sf) = \mathbb{E}\left[e^{-sF}\right] = \mathbb{E}\left[\exp\left(-\sum_{x\in\Phi} sf(x)\right)\right] = \exp\left(-\lambda\int_{R^2}(1-e^{-sf(x)})dx\right) \tag{3.8}$$

where F is a random variable which can be written in the form of $F = \sum_{x\in\Phi} f(x)$.

The PGFL is particularly important in many wireless applications, where the Laplace transform of interference is usually an intermediate step in the characterization of the SINR> Based on the properties of PPP and the Laplace function in (3.8), the statistical behavior of the aggregate interference can be characterized by Laplace function. Assuming that $I(y)$ is the sum of a set of interference who follows the exponential distribution in Φ, then the mean of the aggregate interference can be derived as

$$\mathbb{E}\{\exp(-sI(y))\} = \mathbb{E}\{\exp(-s\sum_{x\in\Phi} l(x-y))\} \tag{3.9}$$
$$= \mathbb{E}\{\prod_{x\in\Phi}\exp\left(-sl(x-y)\right)\}$$
$$= \exp\left(-\lambda\int_{R^2}(1-e^{-sl(x-y)})dx\right)$$

where $x-y$ is the distance between the interfering transmitters to the receiver. Without the loss of generality, it can be assumed that the typical user is located at the origin, the $x-y \to x$ and

$$L_I(s) = \exp\left(-\lambda\int_{R^2}(1-e^{-sl(x)})dx\right) \tag{3.10}$$

Slivnyak's theorem:

The independence between the points of a PPP greatly simplifies the analysis computation. Slivnyak's theorem states that for a Poisson point process Φ, adding or removing a point to a realization, conditioning on a point at x does not change the distribution of the rest of the process.

Slivnyak's theorem is quite simple, but it is important because it allows us to add a node to the PPP at any location we want, such as the origin or at a fixed distance from the origin, without changing its statistical properties. Similarly, it allows removing a small area corresponding to a ball $B(x,\xi)$ for $\xi \to 0$ because the distributions of points in all non-overlapping regions are independent for a PPP.

Properties of Poisson Point Process

- **(Superposition)**: When we set n Poisson point processes $\Phi_1, ..., \Phi_n$, the superposition of these processes is the point process, denoted as $\Phi = \sum_{i=1}^{n} \Phi_i$, whose points are those of $\Phi_1, .., \Phi_n$. In another words, the superposition of n independent Poisson processes of intensities $\lambda_1, ..., \lambda_n$ is a Poisson process of intensity $\lambda = \sum_{i=1}^{n} \lambda_i$.

- **(Independent thinning):** Consider a Poisson point process Φ of intensity λ and a thinning function $p : \mathbb{R}^d \to [0, 1]$. The thinning procedure is considered a realization of a stationary PPP Φ and delete each point with probability $1 - p(x)$, independently of all other points. In the results, an inhomogeneous PPP with density function $\lambda p(x)$ is generated.

  Proof: Let Φ be the original process and $\bar{\Phi}$ the process after thinning. The distribution of $\bar{\Phi}$ is computed as follows:

$$\mathbb{P}\left(\bar{\Phi}(A) = k\right) = \sum_{i=k}^{\infty} \mathbb{P}\left(\Phi(A) = i\right) \mathbb{P}\left(\bar{\Phi}(A) = k | \Phi(A) = i\right)$$

  Given $\Phi(A) = i$, the i points of Φ in A are uniformly distributed. Thus,

$$\mathbb{P}\left(\bar{\Phi}(A) = 1 | \Phi(A) = 1\right) = |A|^{-1} \int_A p(x)dx$$

  The complete conditional distribution is

$$\mathbb{P}\left(\bar{\Phi}(A) = k | \Phi(A) = i\right) = \binom{i}{k} \left(|A|^{-1} \int_A p(x)dx\right)^k \left(1 - |A|^{-1} \int_A p(x)dx\right)^{i-k}$$

  The distribution of $\bar{\Phi}$ is then defined as:

$$\begin{aligned} \mathbb{P}\left(\bar{\Phi}(A) = k\right) &= \sum_{i=k}^{\infty} e^{-\lambda|A|} \frac{(\lambda|A|)^i}{i!} \binom{i}{k} (|A|^{-1}F)^k (1 - |A|^{-1}F)^{i-k} \\ &= e^{-\lambda|A|} \frac{(\lambda F)^k}{k!} \sum_{i=k}^{\infty} \frac{\left(\lambda|A|(1 - |A|^{-1}F)\right)^{i-k}}{(i-k)!} \\ &= e^{-\lambda|A|} \frac{(\lambda F)^k}{k!} e^{\lambda|A|(1-|A|^{-1}F)} \\ &= \frac{(\lambda F)^k}{k!} e^{-\lambda F}, \text{where} \quad F = \int_A p(x)dx. \end{aligned} \tag{3.11}$$

- **(Transformation, displacement theorem)**: If we displace each point of a PPP by some random law, for example, by adding independent and identically distributed (iid) 2D Gaussian random variables to each point, the PP consisting of these new random points will also be PPP.

3.1.3 Applications of Poisson Point Process

In the following, we give some examples of Poisson point process applications:

Mean interference in stationary point process

Let Φ be a stationary point process with intensity λ, $\Phi \subset \mathbb{R}^d$. We assume that each node transmits with unit power and follows the path loss law, which is $l(x) = |x|^{-\alpha}$, where $\alpha > 0$ is a path loss exponent. Then the mean interference power is

$$\mathbb{E}(I) = \mathbb{E}\left(\sum_{x\in\Phi} |x|^{-\alpha}\right)$$

By applying the Campbell's theorem for the mean,

$$\mathbb{E}(I) = \lambda \int_{\mathbb{R}^d} |x|^{-\alpha} dx$$

In the case of two dimensions, i.e. $d = 2$,

$$\lambda \int_{\mathbb{R}^2} |x|^{-\alpha} dx = \lambda \int_0^{2\pi} \int_0^{\infty} r^{-\alpha} r dr d\beta = 2\pi \frac{\lambda}{2-\alpha} r^{2-\alpha}\Big|_0^{\infty}, \quad \alpha \neq 2.$$

For $\alpha = 2$,

$$\lambda \int_{\mathbb{R}^2} |x|^{-2} dx = 2\pi\lambda \log r\Big|_0^{\infty}$$

It can be seen that the mean interference for the power path loss law is infinite in two dimensions for all values of path loss exponent (for $\alpha = 2$, it does not converge anyway).

Interference distribution without fading

Let focus on the case of two-dimensioning networks ($d = 2$) and assume there is no fading, i.e., $I = \sum_{x\in\Phi} g(x)$. Based on isotropic characteristic of $g(x)$, we use $l : \mathbb{R}^+ \mapsto \mathbb{R}^+$, defined by $l(|x|) = g(x)$. The aim is to find the characteristic of interference. First, consider a finite network based on a disk of radius a centered at the origin. The number of nodes k in this area is assumed to be fixed, and their locations are independent and identically distributed (iid).

The interference I_r from the nodes located at a distance a from the origin is defined as

$$I_r = \sum_{x\in\Phi\cap b(o,a)} l(|x|)$$

The characteristic function (Fourier transform) F_{I_a} of interference I_a is

$$\begin{aligned} F_{I_a}(w) &\triangleq \mathbb{E}(e^{jwI_a}) \\ &= \mathbb{E}\left(\mathbb{E}(e^{jwI_a}|\Phi\left(b(o,a)\right) = k)\right) \end{aligned}$$

The probability of finding k nodes in $b(o,a)$ is given by the Poisson distribution, hence:

$$F_{I_a}(w) = \sum_{k=0}^{\infty} \frac{\exp(-\lambda\pi a^2)(\lambda\pi a^2)^k}{k!} \mathbb{E}\left(\mathbb{E}(e^{jwI_a}|\Phi\left(b(o,a)\right) = k)\right)$$

Note that k points in $b(o,a)$ are iid uniformly distributed with radial density

$$f_R(r) = \begin{cases} \frac{2r}{a^2} & \text{if} \quad 0 \le r \le a \\ 0 & \text{otherwise} \end{cases}$$

Therefore, the characteristic function can be calculated as the product of the k individual characteristic functions:

$$\begin{aligned} \mathbb{E}\left(\mathbb{E}(e^{jwI_a}|\Phi\left(b(o,a)\right)=k)\right) &= \left(\int_0^a f_R(r)\exp\left(jwl(r)\right)dr\right)^k \\ &= \left(\int_0^a \frac{2r}{a^2}\exp\left(jwl(r)\right)dr\right)^k \end{aligned}$$

By interpreting the summing over k as the Taylor expansion of the exponential function, we obtain

$$F_{I_a}(w) = \exp\left(\lambda\pi a^2\left(-1+\int_0^a \frac{2r}{a^2}\exp(jwl(r))dr\right)\right)$$

In the limit $a \to \infty$, $I_a \to I$, after integration by part, we have

$$F_I(w) = \exp\left(j\lambda\pi w\int_0^\infty (l^{-1}(x))^2 e^{jwx}dx\right)$$

where l^{-1} is the inverse of l, $l^{-1}(x) \leftarrow r$, and

$$\lim_{a\to\infty} a^2\left(-1+\int_0^a \frac{2r}{a^2}\exp(jwl(r))dr\right) = \int_0^\infty (l^{-1}(x))^2 jwe^{jwx}dx$$

In particular, for the power law $l(w) = r^{-\alpha}$, we obtain

$$\begin{aligned} F_I(w) &= \exp\left(j\lambda\pi w\int_0^\infty x^{-2/\alpha}e^{jwx}dx\right) \\ &= \exp\left(-\lambda\pi\Gamma(1-2/\alpha)w^{2/\alpha}e^{-j\pi/\alpha}\right), \quad w \ge 0, \alpha > 2 \end{aligned}$$

In the case of $\alpha \le 2$, the integral diverges, and the interference is infinite. The above characteristic function $F_I(w)$ indicates that the interference distribution is a stable distribution with characteristic exponent $2/\alpha < 1$. The corresponding Laplace transform is

$$L_I(s) = \exp\left(-\lambda\pi\Gamma(1-2/\alpha)s^{2/\alpha}\right)$$

We will consider the case of interference distribution and outage in the NB-IoT network with Rayleigh fading in the next Chapter.

Poisson-Voronoi tessellation

Given a collection of points $\Phi = X_i$ and a given point x, the Voronoi cell $C_x(\Phi)$ of this point is defined as the subset of the plane of all locations that are closer to x than to any point in Φ:

$$C_x(\Phi) = \{y \in \mathbb{R}^d : ||y-x|| < \inf_{x_i\in\Phi, x_i\neq x} ||y-x_i||\}$$

The collection of cells $\{C_{xi}(\Phi)\}$ is called the Poisson Voronoi tessellation when $\Phi = \{X_i\}$ is a Poisson point process. An example of Poisson Voronoi tessellation is given in Figure 3.1.

Assume that the receiver is associated with the nearest transmitter with density λ, and using the fact that the null probability of a 2-D Poisson process in an area A is $\exp(\lambda A)$:

$$P[r > R] = e^{-\lambda \pi R^2}$$

Therefore, the probability density function (pdf) of the distance r between the transmitter and the receiver can be expressed as

$$f_{R(r)} = \frac{dF_r(R)}{dr} = \frac{d(P[r \leq R])}{dr} = 2\pi\lambda r\, e^{-\lambda \pi r^2}$$

To conclude, using mathematical tools from stochastic process to derive the theoretical expressions of network performance metrics, is worth to be considered in a random distributed system model.

3.2 Machine Learning Background

As can be seen, the application of Machine learning (ML) in future networks has limitless potential. Due to their inherited property of analyzing data, build models and make future predictions, ML is expected to play a major role in future mobile networks. It will make future networks more reliable, efficient, cost-effective, and manageable. Machine learning aims to take input data and learn a model based on a specific set of algorithms that relates the given input with the desired output. Machine learning is widely applied to several critical problems such as natural language processing, pattern recognition, speech and handwriting recognition in computer vision or image processing fields.

In this Section, we present an overview of the algorithms of Machine Learning. First, we give a broad categorization of the ML algorithms based on the learning procedure, and then we introduce the key principles of Deep learning algorithms. The advantages and limitations of Deep learning in Wireless networks are also derived at the end of this Section.

3.2.1 Machine learning categories

The ML algorithms can be broadly partitioned in supervised, unsupervised, and reinforcement learning algorithms based on the learning process. The taxonomy of machine learning algorithms are illustrated in Figure 3.2.

Supervised Learning

Supervised learning is a machine learning task that aims to learn a mapping function from the input to the output, given a labeled data set. The algorithm learning process from the training dataset can be considered as a teacher supervising the learning process. The algorithm iteratively makes predictions on the training data and is corrected based on knowing the real value of output. The learning procedure stops when the algorithm achieves acceptable performance. Specifically, supervised learning can be further divided into regression and classification based on the specific requirements. There are several supervised learning techniques:

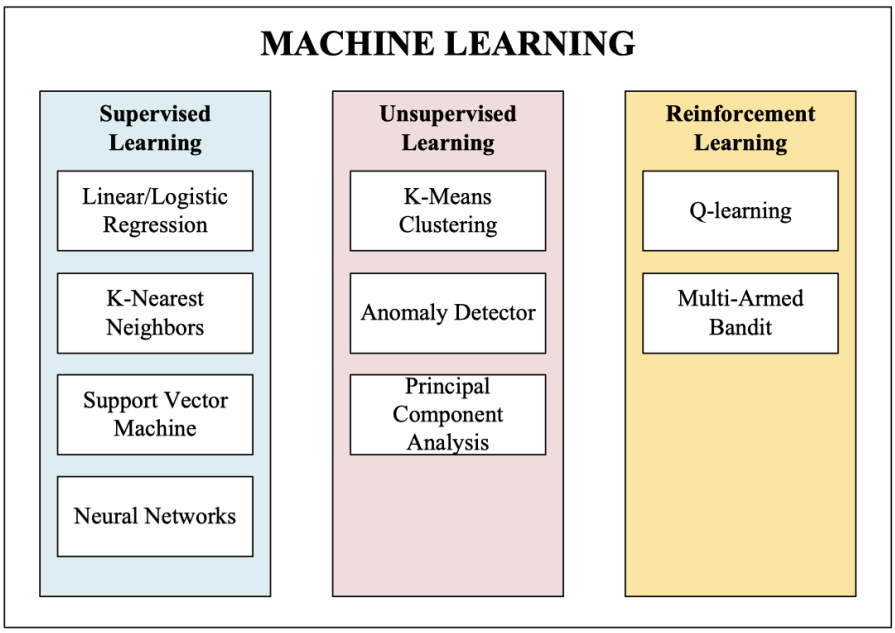


Figure 3.2: Taxonomy of Machine Learning Algorithms.

- *Linear Regression*: In linear regression, the aim is to learn a function $f(x, w)$. It is a basic function, which is a linear combination of a fixed set of linear or non-linear functions of the input variables $\phi_i(x)$ and can be described as:

$$f(x, w) = \phi(x)^T w, \tag{3.12}$$

  where w is the weight vector or matrix $w = (w_1, ..., w_D)^T$, and $\phi = (\phi_1, ..., \phi_D)^T$. There exist a large number of basic functions, such as sigmoidal, polynomial, Gaussian functions, which can be chosen with respect to the different aims. By using suitable basic functions, it can be shown that the mapping from the input variable to the output variable can be modeled by arbitrary non-linearities.

  There are many approaches for training the model, such as Ordinary Least Square, Regularized Least Squares, Least-Mean-Squares (LMS), or Bayesian Linear Regression. Among them, LMS is widely used because it can learn the parameters online by applying the technique of stochastic gradient descent.

- *Support Vector Machine*: The basic of the support vector machine (SVM) model is a binary classifier, aiming to find the dividing hyperplane that separates both classes of the training set with the maximum margin. The maximum-margin hyperplane is the best hyperplane that leads to the largest seperation between two given classes. The predicted label of an unseen data point is then determined based on which side of the hyperplane it falls. The binary linear classification task can be described using a

constrained optimization problem:

$$\begin{aligned} \underset{w,b,\xi}{\text{minimize}} \quad & f(w,b,\xi) = \frac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i\left(w^T x_i + b\right) - 1 + \xi_i \geq 0 \quad i = 1,2,...,n \end{aligned} \tag{3.13}$$

where w denotes the normal vector to the hyperplane, b is the parameter for controlling the offset of the hyperplane from the origin along its normal vector, ξ_i denotes a slack variable which gives the distance by which every training point x_i violates the margin in the units of $|w|$ and parameter C determines how heavily a violation is punished, $C > 0$. The parameter C can be chosen via cross-validation or Bayesian optimization.

Besides, SVMs can perform a non-linear classification, which finds a hyperplane that is a non-linear function of the input variable. In this case, the original space can be mapped to high-dimensional feature spaces by involving kernel functions, such as polynomial or Gaussian kernels.

In addition, the SVM can be extended to solve the regression problems. The result of SVM regression model depends only on a subset of the training points, owning to the rejection of training points that are close to the model prediction.

To conclude, SVMs are one of the most supervised learning models that are capable of effectively dealing with high-dimensional datasets and are efficient in terms of memory usage. They are widely used in many real-world applications such as hand-written character recognition or image classification. One significant limitation of SVM is that it does not directly provide probability estimates.

- *K Nearest Neighbors*: K Nearest Neighbors (KNN) is a non-parametric learning algorithm for classification and regression, where no assumption on the data distribution is needed. For example, the objective in the classification task is to classify a new given unseen data point by looking at the K given data points in the training set that are closest to it in the input space. In other words, the process is based on the majority voting of its K nearest neighbors using a distance metric, such as Euclidean distance, L_{inf}, or Hamming distance. Therefore, the form of classification task is described as

$$\begin{aligned} p(t = c|x, K) &= \frac{1}{K}\sum_{i \in N_k(x)} 1(t_i = c) \\ y &= \arg\max p(t = c|x, K) \end{aligned} \tag{3.14}$$

  where x is the new input data point, $N_k(x)$ is the K nearest neighbors of x, y denotes the predicted class label for x, t is a discrete random variable, and $1(x)$ denotes the indicator function.

  One drawback of KNN is that it requires storing the entire training set, which makes KNN unscalable to large datasets.

Unsupervised Learning

Unsupervised learning is a machine learning task that aims to learn a function to describe a hidden structure from unlabeled data. The following unsupervised learning techniques are utilized:

- *K-Means Clustering Algorithm*: In K-means clustering, the objective is to partition the unlabeled n data points of x into K clusters, where data points belonging to the same cluster must with nearest mean and have some similarities. The most common version of K-means algorithms is based on iterative refinement. In the beginning, K-means are randomly initialized. Then, in each iteration, each data point is assigned to exactly one cluster, which has centers in a set of K cluster centers, denoted by $\{s_1, ..., s_k\}$. The mean of each cluster is updated on the condition that the distances between the data points and their nearest center must be minimized. The algorithm continuously iterates until the members of each cluster do not change. The K-means algorithm is described as following

$$\begin{aligned} \underset{s,\pi}{\text{minimize}} \quad & \sum_{n=1}^{N}\sum_{k=1}^{K} \pi_{nk}\|x_n - s_k\|^2 \\ \text{subject to} \quad & \sum_{k=1}^{K} \pi_{nk} = 1, n = 1, ..., N \end{aligned} \tag{3.15}$$

  The data point x_n is assigned to the cluster center s_k if its binary indicator variables $\pi_{nk} = 1$, where $\pi_{nk} = \{0, 1\}$.

  Besides the advantage of the very fast and high scalable algorithm, the K-means clustering has several limitations. This algorithm assigns each data point only one of the clusters, which may lead to inappropriate clusters in some cases. In addition, the cluster centers are not robust against outliers.

- *Principal Component Analysis*: Principal component analysis (PCA) is a dimension-reduction tool that can be used to reduce a large set of variables to a small set that still contains most of the information. The objective of PCA is finding an orthogonal linear transformation of x_n in which the output is a set of L linear M-dimensional basic vectors $\{w_j\}$ and the corresponding linear projections of data points $\{z_{nj}\}$. The result is satisfied with the average reconstruction error J is minimized.

$$\begin{aligned} J &= \frac{1}{N}\sum_{n} \|\tilde{x}_n - x_n\|^2 \\ \tilde{x}_n &= \sum_{j=1}^{L} z_{nj} w_j + \bar{x} \end{aligned} \tag{3.16}$$

  where $\bar{x}$ is the average of all data points.

  PCA is one of the most important preprocessing techniques in machine learning. Its application involves data compression, whitening, and data visualizations. Examples of its practical applications are face recognition and neuroscience.

Reinforcement Learning

In reinforcement learning (RL), the agent learns a policy to optimize a long-term objective by interacting with the unpredictable environment. Every action from the agent has some impact on the environment, and the environment provides feedback (reward) that guides the learning algorithm. The RL model where an agent is interacting with the environment is illustrated

in Figure 3.3. A list of examples includes Q-learning, Multi-Armed Bandit Learning (MAB), etc.

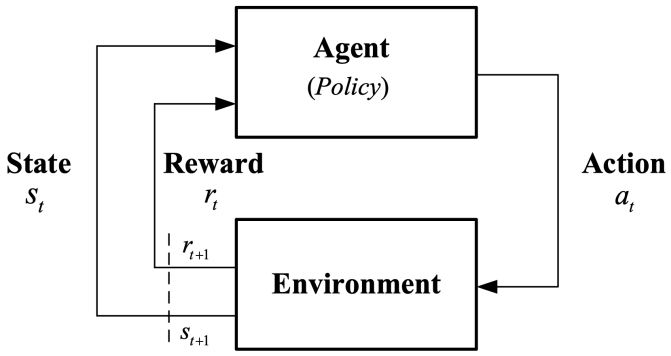


Figure 3.3: Reinforcement Learning Framework.

- *Q-Learning*: One of the most popular algorithms in RL is Q-learning. The agent interacts with the environment to learn the Q values, based on which the agent takes an action. The Q-value is defined as the discounted accumulative reward, starting at a tuple of a state and an action and then following a certain policy. The agent can make a quick decision under the current state by taking action with the largest Q value. More details about Q learning can be referred to [17].

- *Multi-Armed Bandit Learning*: In a multi-armed bandit (MAB) learning with a single agent, the agent sequentially takes an action and then receives a random reward generated by a corresponding distribution, aiming at maximizing an aggregate reward. There exists a tradeoff between taking the current, best action (exploitation) and gathering information to achieve a larger reward in the future (exploration). While in the MAB with multiple agents, the received reward after giving action is not only dependent on this action but also on the agents taking the same action. In this case, the model is expected to achieve some steady states or equilibrium. More details about MAB can be referred to [18].

This thesis focuses on the modeling for predicting network performance and mobile network traffic characterization rather than optimizing network procedures. Therefore, RL has not been used in the solutions presented in this work.

3.2.2 Deep Learning

Deep learning is essentially a sub-brand of ML, which enables an algorithm to make predictions, classifications, or decisions based on data, without explicitly programmed. The classic algorithms include linear regression, the K-nearest neighbors classifier, and Q-learning, which are introduced in the previous section. These algorithms extract knowledge from raw data through multiple layers of nonlinear processing units in order to make predictions or actions according to the different objectives. The most well-known deep learning models are neural networks (NNs). Neural networks are organized hierarchically in layers of processing units

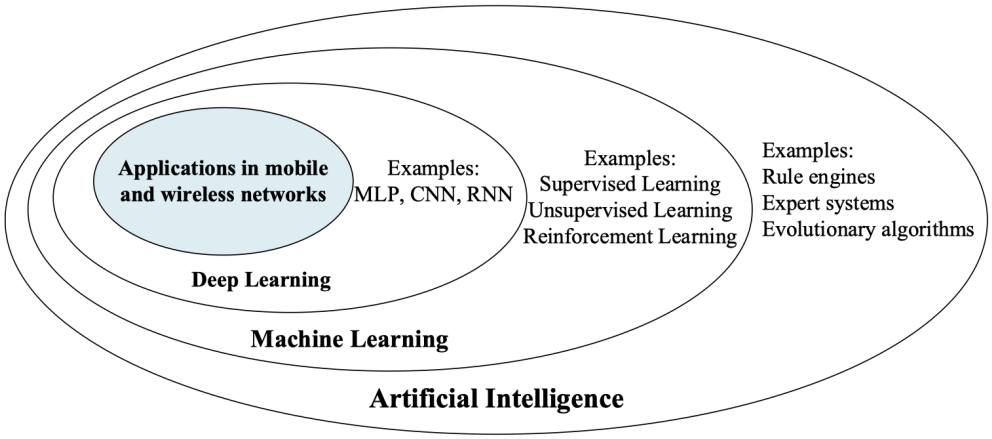


Figure 3.4: The relationship between deep learning, machine learning, and AI.

characterized by an input layer, one or more hidden layers, an output layer, and weights and bias terms. If the network has only one hidden layer, it is called a shallow neural network, whereas if there is more than one hidden layer, it is referred to as a deep neural network. Broad examples of deep learning include Multi-layer perceptron (MLP), Convolutional Neural Network (CNN), and Recurrent Neural Network (RNN). Figure 3.4 gives a relationship between deep learning, machine learning, and AI.

In the following, we introduce the fundamental principles of MLP and RNN models, which are used in this thesis. The more detail of MLP and RNN can continue reading in Chapter 4 and Chapter 6, respectively.

Multilayer Perceptron

Similar to the brain cells, a neuron is the basic element of Artificial Neural Network (ANN). In practice, the simplest neuron is represented by the Perceptron, which performs basic mathematical operations. Perceptrons can be combined to form a layer architecture, and layers can be connected to build multilayer networks, where the nodes are the neurons, and the edges are connections.

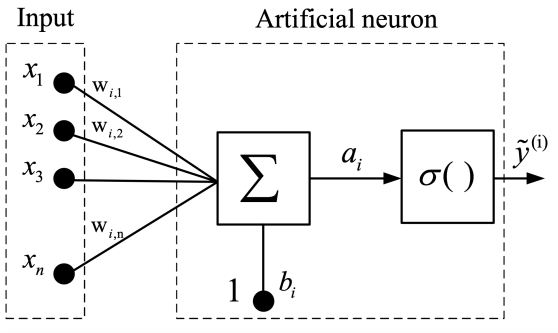


Figure 3.5: Perceptron representation.

Supposes our dataset consists of m input-output pairs $(x^{(i)}, y^{(i)}), i = 1, ..., m$. The simplest

perceptron takes a vector of input $x^{(i)} = [x_1, x_2, ..., x_n]$ and produces an output $\tilde{y}^{(i)}$ as shown is Figure 3.5. It performs a linear combination with the weights w and non-linear activation function $\sigma(.)$. The complete equation for a neuron can be written as

$$\tilde{y}^{(i)} = \sigma\left(\sum_{j}^{n} x_j w_{i,j} + b_i\right) \tag{3.17}$$

Generally, the single neuron operation can be mathematically expressed as matrix multiplication. The notation in 3.17 simplifies as follows:

$$\tilde{y}^{(i)} = \sigma\left(w^T x^{(i)}\right), \tag{3.18}$$

where $x^{(i)} = [1, x_1, x_2, ..., x_n]$ and the vector of weights $w = [b, w_1, w_2, ..., w_n]$.

A multi-layer perceptron is a combination of multiple layers. Each neuron in the different layers performs the same computation as shown in Equation (3.18). The first and the last layers of the neural network are the input and the output, respectively. The intermediate layers are called hidden layers; the neurons of this layer defines the size of the layer. An example of MLP architecture with one hidden layer and five hidden neurons is represented in Figure 3.6

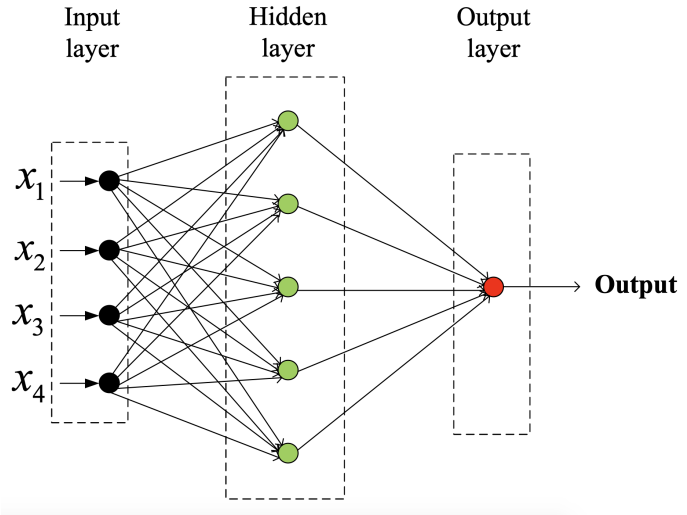


Figure 3.6: MLP architecture.

The activation function introduces non-linearity into the output of a neuron. There are a number of activation functions that introduced in the literature, like *a sigmoid*, *hyperbolic tangent*, or *a rectified linear unit*. Table 3.1 gives a comparison between several activation functions used in a regression problem. The mathematical explanation of these functions are explained below:

- Sigmoid:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.19}$$

- Hyperbolic tangent:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.20}$$

- Rectified Linear Unit (ReLu):

$$f(x) = \max(0, x) \tag{3.21}$$

The weights of the neural network are trained using the *backpropagation* technique. The idea of this technique is to start with random initialization of the weights and calculate the output for a given input. The error between the generated output and the actual output is then calculated and used to update the weights of the network using a gradient descent algorithm. More details on the operation of the backpropagation algorithm are provided in Chapter 4.

Recurrent Neural Networks

Recurrent neural networks (RNNs) are a family of neural networks for processing sequential data. There are various applications of RNNs, such as speech recognition, language processing, image recognition, etc. The typical structure of a RNN is illustrated in Figure 3.7, which consists of a neural network with loops that allow processing the information. In the diagram, a neural network A takes x_t as the input and outputs a value h_t.

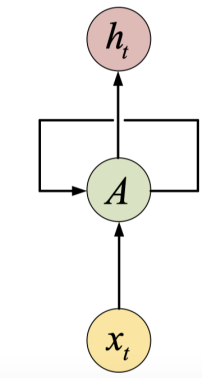


Figure 3.7: A RNN structure.

The objective of RNN architecture is to exploit the sequential structure of the data. The unrolled RNN structure is illustrated in Figure 3.8. The same operation A is performed for every element of sequence. The output h_t depends on the current input x_t and the previous operations which contain information about what happened in the network up to time t.

3.2.3 Advantages of Deep Learning in Mobile and Wireless Networks

There are several benefits of employing deep learning to solve network engineering problems, such as:

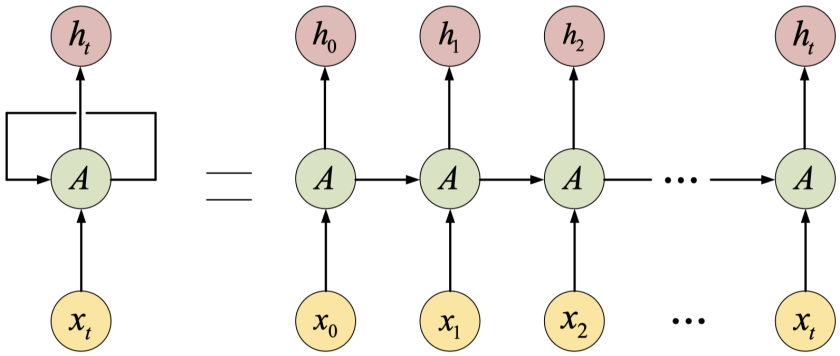


Figure 3.8: Unrolled RNN structure.

- Applying deep learning techniques can automatically extract high-level features from data that has a complex structure and inner correlations. The learning process does not need to be designed by a human, which tremendously simplifies prior data pre-processing. This is very important in the context of mobile networks because mobile data is usually generated by heterogeneous sources. This data is often noisy and exhibits non-trivial spatial-temporal patterns.

- Deep learning is capable of handling large amounts of data. In fact, mobile networks generate high volumes of different types of data. Traditional machine learning algorithms sometimes require storing all the data in memory, which, lead to infeasible computation in big data scenarios. Deep learning algorithms only require sub-sets of data at each training step, which guarantees the scalability with big data. Therefore they also further prevent model over-fitting.

- Compressive representations learned by deep neural networks can be shared across different tasks, while this is limited or difficult to achieve in other ML architectures. Therefore, a single model can be trained to adapt multiple objectives without requiring complete model retraining for different tasks. It reduces computational and memory requirements of mobile systems when performing multitask learning applications.

3.2.4 Limitations of Deep Learning in Mobile and Wireless Networks

Deep learning has several limitations, which partially restrict its applicability in the mobile networking domain. In the following, we report some of the main limitations:

- There are many black boxes in deep learning algorithms and lead to having low interpretability. In other words, deep learning enables creating systems that have high accuracy in specific tasks. But we still have limited knowledge of why making the decisions. Therefore in some cases, researchers would rather apply statistical methods to have high interpretability. In addition, in mobile networks, it is difficult to obtain sufficient information for model training. The data collection may be costly and face privacy concerns. In such scenarios, the benefits of employing deep learning may not be cost-effective.

- There is a large number of demanding computations in deep learning. Deep neural networks usually require complex structures to achieve satisfactory accuracy performance. However, there is a difficulty when employing neural networks on embedded and mobile devices due to energy and capability constraints.
- Deep neural networks usually have many hyper-parameters, and finding their optimal configuration can be difficult. The number of hyper-parameters grows exponentially with the depth of the model and can highly influence its performance.

3.3 Summary

In this Chapter, we discuss the mathematical tools used and the Machine Learning framework applied to wireless network analysis. First, we have presented the Poisson point process definition and its properties. After, we have introduced the applications of the Poisson point process in wireless networks. Then we have present the Machine learning background and reviewed the key principles of Deep learning models, which are used in the following chapters. The advantages and limitations of deep learning in mobile and wireless networks are also drawn in this Chapter.

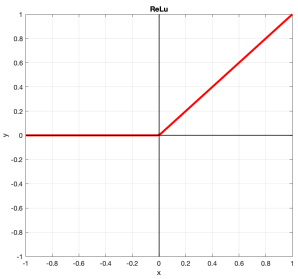
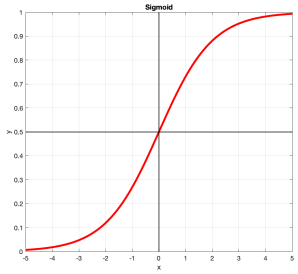
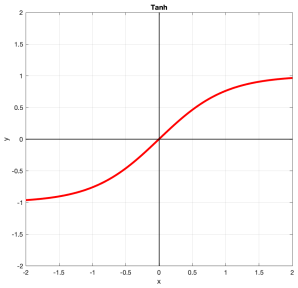
Activation functions	Advantages	Disadvantages
ReLu 	• Avoids and rectifies vanishing gradient problem; • ReLu is less computationally expensive than tanh and sigmoid.	• Should only be used within hidden layers of a NN model; • For activations in the region ($x < 0$) of ReLu, gradient will be 0. It is also called dying ReLu problem; • The range of ReLu is $[0, inf]$.
Sigmoid 	• The sigmoid is nonlinear in nature and the combinations of this function are also non-linear; • It has a smooth gradient; • The activation bound in a range, its output is always going to be in range $(0, 1)$.	• The Y values tend to respond very little to changes in X toward the end; • It raises a problem of "vanishing gradients";
Tanh 	• The gradient is stronger for tanh than sigmoid.	• Tanh also has the vanishing gradient problem.

Table 3.1: Activation functions

Chapter 4

Performance Evaluation of Coverage in NB-IoT Networks

4.1 Introduction

Providing connectivity for massive Internet-of-Things (IoT) devices is a key driver of 5G. In IoT use cases, an IoT device must be reachable even if it is deployed in basements or deep-indoor environments. This leads to the need to extend coverage as well as satisfying the required battery lifetime. There are several proposed solutions for enabling large-scale IoT connectivity, including evolutionary and revolutionary solutions. NB-IoT has been announced as a revolutionary solution that handles communication over a 200 kHz bandwidth [19]. This narrow bandwidth brings a high link budget and offers extended coverage. NB-IoT targets devices with poor Signal to Noise Ratio (SINR) conditions. NB-IoT provides substantially enhanced coverage, which is a significant improvement compared with legacy LTE technologies, especially to serve the users in deep coverage. It is very necessary to study how much coverage NB-IoT can actually provide.

This chapter analyzes the coverage performance of NB-IoT in single-cell and multi-cell networks. In our analysis, we assume the UEs are located according to a homogeneous Poisson point process and compute the uplink SINR for two main cases. We first derive an analytical expression based on stochastic geometry to study the coverage and the successful transmission probability performances. We assume the location of sensors is modeled as a realization of PPP, and the eNB is located at the origin. The analytical expressions of the coverage and successful access probabilities are derived by taking into account the packet arrival distribution. The analysis includes the limitations due to only considering the single-cell NB-IoT network. We adopt the repetitions in NB-IoT to study the access success probability. Next, we used the machine learning approach to derive the model that can predict the coverage performance of a multi-cell NB-IoT network based on the input network parameters without considering the interference characterization.

The rest of the Chapter is organized as follows. Section 4.2 provides the related works on NB-IoT coverage analysis. Section 4.3 presents the analytical model for the performance evaluation of the uplink transmission in NB-IoT networks. Section 4.4 describes the coverage analysis in a multi-cell NB-IoT network based on the neural networks. The comparison between applying different supervised learning algorithms is drawn in the evaluation. Lastly, section 4.5 presents the summary of this chapter.

4.2 Related works

Recent works on NB-IoT provide insight on different issues related to the coverage performance. In [20], *Lauridsen et al.* analyze the coverage and capacity performance of LTE-M and NB-IoT in a rural area. They use the commercial LTE configuration and location and calibrate the simulation using drive test measurements. The results show that NB-IoT can provide coverage for more than 95% of basement or deep indoor devices, which is better than 80% of LTE-M. This work also shows that the cost of providing deep indoor coverage with NB-IoT is not only a lower number of supported devices per sector due to the large Radio Resource Control delay overheads, but also 2-6 times higher device consumption as compared to LTE-M.

In [21], *Adhikary et al.* provide a comprehensive evaluation of the coverage performance of NB-IoT system. Their evaluation is based on the assumption of a single cell, and inter-cell interference arising from multiple transmitting cells in the network is not considered. The results show that NB-IoT provides excellent coverage of up to 20 dB when compared with the legacy LTE system. Additionally, they prove that NB-IoT provides good co-existence performance with the existing LTE system.

In [22], *Ali et al.* present an analytical signal model and NDMRS generation and mapping. Two channel estimation algorithms as a modification of traditional LS and MMSE estimators are proposed to guarantee the successful detection of user data in extremely low SNR. The proposed techniques provide better estimation precision compared to the conventional algorithms and can be adopted to NB-IoT uplink receiver. In addition, they analyze and evaluate the PAPR by employing RC and RRC pulse shaping at the transmitter. The results show that the RRC pulse shaping with lower PAPR values is feasible to the actual hardware design of low-cost NB-IoT UE.

In [23], the authors provide a comparison in terms of coverage and capacity between SigFox, LoRa, GPRS, and NB-IoT in a real deployment scenario in Denmark. They simulate the link loss between the UEs and the base stations and compare it with the MCL of each technology. The results show all four technologies provide better than 99% outdoor coverage. For indoor coverage, NB-IoT outperforms the other technologies [24]. However, NB-IoT may experience significant interference from legacy LTE systems due to utilizing the same resources, while SigFox may encounter capacity problems.

Chafii et al. in [25] propose a new method to enhance the coverage based on machine learning. The proposed method based on the dynamic spectrum using reinforcement learning algorithms can help to increase the coverage, decreasing the number of repetitions and reducing the energy consumption.

In addition, stochastic geometry has been regarded as a powerful mathematical tool for modeling, analyzing, and designing wireless networks with random topologies [26]. It has been applied to characterize the distribution of interference and the outage probability in ad-hoc networks. In particular, some models investigate the performance of random channel access based on ALOHA [27, 28, 29].

Many works have already investigated the performance of the uplink of cellular systems [6, 30, 31]. The coverage probability and average rate of a typical uplink in the cellular networks were successfully obtained in [30]. The proposed models are applicable both to uniform and irregular network topologies. In [31], the authors analyze the coverage probability of multi-user uplink cellular network with fractional power control, where the Poisson Point process was used to model mobile user locations.

As can be seen, recent works in the literature address the coverage performance of NB-IoT networks and provide an insight on this topic. However, there is a limit to the application of stochastic geometry and machine learning to the coverage analysis of the NB-IoT networks. The analysis presented in this chapter is complementary to the existing literature and joins the analysis of the challenges when extending the analysis for multi-cell scenarios.

4.3 Coverage analysis in a single-cell NB-IoT network

4.3.1 System model

The Random Access (RA) procedure has the same message flow as for LTE, however, with different parameters as illustrated in Chapter 2. For NB-IoT, the RA procedure is always contention-based. Recall that if the associated Random Access Response (RAR) message was not received in the second step, the UE transmits another one which can be repeated up to N_r times, where $N_r = 2^q$, $q = 0, ...7$. For the case that this maximum number of attempts is reached without successful transmission, the UE proceeds to the next coverage level. NB-IoT networks can support up to three coverage levels, which can be configured with different value of N_r. If the maximal number of access attempts is reached, a failure is reported to the Radio Resource Control (RRC).

Model and Parameters

Figure 4.1 illustrates the uplink system model and the relationship between parameters. The key aspects of this model are as follows:

- eNB is located at the origin of the Euclidean plane. The eNB has a single active uplink scheduled on a given time-frequency resource which is randomly chosen from all UEs in the cell.

- User Equipments (UEs) are located according to a homogeneous PPP Φ_u of intensity λ_n. This is how cellular networks are generally simulated in practice by using a uniform distribution of users (which is equivalent to a PPP in a given area conditioned on the number of users). The UEs constantly transmit with fixed power p to the eNB on any particular time-frequency resource, i.e. orthogonal multiple access within a cell. An UE randomly selects one preamble before transmission. Two sensors generate a collision if they select the same preamble if no capture effect occurs. Therefore, the interference comes from other devices transmitting in uplink inside the cell. We denote the set of interfering devices by $Z = \{z_i\}, i = 1, 2..n$, the distance from an interfering device $z_i \in Z$ to the eNB by D_i.

- The uplink signals attenuate with distance according to the standard power-law path loss propagation model, which is inversely proportional to distance $r^{-\alpha}$, where $\alpha > 2$ is the path loss exponent, and r is the distance from the transmitting node to the receiving node. Specifically, we assume that the received power at distance r is $P_r = pr^{-\alpha}$.

- Random channel effects are incorporated by a multiplicative random value g for the desired signal and g_i for interferer i. For simplicity, we assume these all correspond to small-scale Rayleigh fading with mean 1, i.e., g and g_i are iid and follow an exponential

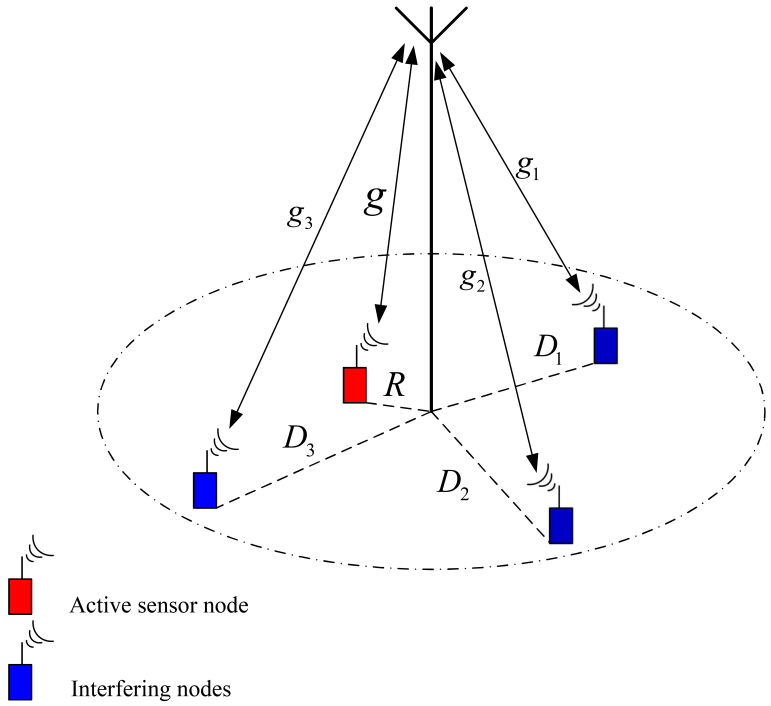


Figure 4.1: NB-IoT uplink system model.

distribution with unit mean $\mathcal{E}(1)$. We note that more general fading distributions can be accommodated but with a loss of tractability and without much change to the results and system design insights [32].

- We consider the uplink SINR, which is computed for the nearest active UE connected to the eNB.

Under this system model, if the distance between the NB-IoT device and the eNB is R then the received power at the eNB is $P_r = gpR^{-\alpha}$ and the uplink SINR at a eNB located at the origin is:

$$\text{SINR} = \frac{g\, p\, R^{-\alpha}}{\sigma^2 + I_Z}. \tag{4.1}$$

where σ^2 models the thermal noise power and I_Z denotes the interference created by the set of interfering devices Z:

$$I_Z = \sum_{z_i \in Z} (p\, g_i\, D_i^{-\alpha}) \tag{4.2}$$

Distance to the Nearest UE.

An important quantity is the distance R from a typical UE to the eNB. Since we consider the closest UE, no other UE can be closer than R, so all interfering UEs must be farther than R.

The probability density function (PDF) of R can be derived using the null probability of a 2-D Poisson process in an area A is $\exp(-\lambda A)$.

$$\mathrm{P}[R > r] = \mathrm{P}[\text{No eNB closer than} r] = e^{-\lambda \pi r^2} \tag{4.3}$$

As presented in [33], distance R is assumed to be i.i.d. random variable following Rayleigh distribution with probability density function of R can be found as:

$$f_R(r) = \frac{dF_R(r)}{dr} = 2\pi\lambda r e^{-\lambda\pi r^2}, r \geq 0 \tag{4.4}$$

where $F_R(r)$ is the cumulative distribution function (CDF) of R and is defined as $F_R(r) = \mathrm{P}[R \leq r] = 1 - e^{-\lambda\pi r^2}$.

The main goal of the next sections is to provide an analytical expression that gives a close approximation for the SINR distribution.

4.3.2 Analytical model for coverage probability

The successful access probability

Assume that the channel transmission attempts at eNB follow a Poisson distribution and that the probability of selecting a preamble follows an uniform distribution. A collision occurs when more than one sensor transmits at the same time with the same preamble [34]. Based on Poisson distribution, the collision probability p_f is then equal:

$$p_f = 1 - \frac{\sum_{i=1} i \frac{(\lambda_{Total}/N_{ra})^i}{i!} \exp(-\lambda_{Total}/N_{ra})}{\lambda_{Total}/N_{ra}} \tag{4.5}$$

$$= 1 - \exp\left(-\frac{\lambda_{Total}}{N_{ra}}\right) = 1 - p_s, \tag{4.6}$$

where λ_{Total} is the total arrival rate, λ is the intensity of new arrivals, N_{ra} is the number of random access preamble, and p_s is the successful access probability.

It can be assumed that the device retransmits RA preamble at the NPRACH(j) if it has a collision at the NPRACH($j-1$), we have $\lambda_{Total}(j) = \lambda + p_f.\lambda_{Total}(j-1)$. When a system is in a steady-state, it means $\lambda_{Total}(j) = \lambda_{Total}(j-1)$ for all j, the total arrival rate can be defined as

$$\lambda_{Total} = \lambda / p_s. \tag{4.7}$$

We assume that each packet experiences channel outage probability, which is defined as the probability that the SINR at the input of the receiver is falling below a given threshold value T. Mathematically, the outage probability can be obtained from the CDF of the SINR and given by P(SINR $\leq T$). In addition, to extend coverage, a repetition transmission scheme has been applied to the NB-IoT system. The maximum number of repetitions for an uplink transmission has been defined as 128 in the system. From that, the successful access probability is:

$$p_s = e^{-\lambda_{Total}/N_{ra}}.\left(1 - \mathrm{P}(\mathrm{SINR} < T)^{N_r}\right)$$

$$= e^{-\lambda_{Total}/N_{ra}}.(1 - (1 - p_c)^{N_r}), \tag{4.8}$$

where N_r is the number of repetitions and $p_c = \mathrm{P}(\mathrm{SINR} \geq T)$ is the coverage probability.

By substituting (4.8) to (4.7), we can obtain (4.9)

$$\lambda_{Total}.e^{-\lambda_{Total}/N_{ra}}.Q = \lambda, \tag{4.9}$$

where $Q = (1-(1-p_c)^{N_r})$. By using the Lambert W-function [], the total arrival rate can be derived as

$$\lambda_{Total} = -N_{ra}.W_0.\left(-\lambda/(N_{ra}.Q)\right), \tag{4.10}$$

where W_0 indicates the Lambert W-function, i.e. the inverse function of $f(x) = xex$.

Interference Characterization

The interference at the tagged eNB is the sum of powers from all the interfering UEs, which are modeled by the PPP. This power depends on the distance from the UE to the tagged eNB. To characterize the interference, we compute the Laplace transform of random variable I_Z, which we denote as L_{I_Z}. The Laplace transform definition yields [35]

$$\begin{aligned} L_{I_Z}(s) &= \mathbb{E}_{I_Z}[e^{-sI_Z}] \\ &= \mathbb{E}_{I_Z}\left[\exp\left(-\sum spg_iD_i^{-\alpha}\right)\right] \\ &= \mathbb{E}_{g_i,D_i}\left[\prod \exp\left(-spg_iD_i^{-\alpha}\right)\right] \end{aligned} \tag{4.11}$$

Using the independence of g_i, we can move the expectation with respect to g_i inside the multiplication:

$$L_{I_Z}(s) = \mathbb{E}_{D_i}\left[\prod_{z_i\in Z} \mathbb{E}_{g_i}\left(\exp\left(-spg_iD_i^{-\alpha}\right)\right)\right] \tag{4.12}$$

Definition 1 (Moment generating function of exponential distribution): A continuous random variable X is said to have an exponential distribution with parameter $\beta > 0$, denoted as $X \sim \mathcal{E}(\beta)$, if its PDF is given by:

$$f_X(x) = \begin{cases} \beta e^{-\beta x}, & x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.13}$$

We say that X processes a moment generating function, and the function $M_X(t) = \mathbb{E}[\exp(tX)]$ is called **the moment generating function** of X.

$$\begin{aligned} \mathbb{E}[\exp(tX)] &= \int_{-\infty}^{\infty} \exp(tx) f_X(x)dx \\ &= \int_0^{\infty} \exp(tx)\beta\exp(-\beta x)dx \\ &= \beta\int_0^{\infty} e^{(t-\beta)x}dx \\ &= \beta\left[\frac{1}{t-\beta}e^{(t-\beta)x}\right]_0^{\infty} = \frac{\beta}{\beta-t} \end{aligned} \tag{4.14}$$

After changing the variable $D_i = \mathbf{x}$ and using the PGFL of PPP with respect to the function $f(\mathbf{x}) = \mathbb{E}_{g_i}\left(\exp\left(-spg_i\mathbf{x}^{-\alpha}\right)\right)$, we get

$$L_{I_Z}(s) = \exp\left(-\lambda_n \int \left(1 - \mathbb{E}_{g_i}\left(\exp(-spg_i\mathbf{x}^{-\alpha})\right) \mathrm{d}\mathbf{x}\right)\right) \tag{4.15}$$

Employing a transformation to polar coordinates $\mathbf{x} = (x, \theta)$, we get

$$L_{I_Z}(s) = \exp\left[-2\pi\lambda_n \int_r^\infty \left(1 - \mathbb{E}_{g_i}\left(\exp\left(-spg_i x^{-\alpha}\right)\right)\right) x\mathrm{d}x\right] \tag{4.16}$$

From the result of the moment generating function (4.14) and the assumption $g_i \sim \mathcal{E}(1)$, we have

$$\begin{aligned} L_{I_Z}(s) &= exp\left[-2\pi\lambda_n \int_r^\infty \left(1 - \left[\frac{1}{1+(sp)x^{-\alpha}}\right]\right) x\mathrm{d}x\right] \\ &= exp\left[-2\pi\lambda_n \int_r^\infty \left[\frac{1}{1+(sp)^{-1}x^{\alpha}}\right] x\mathrm{d}x\right] \end{aligned} \tag{4.17}$$

The coverage probability

The uplink coverage probability is defined as the complementary cumulative distribution function (CCDF) of uplink SINR, i.e., the probability that the uplink SINR at the tagged eNB is greater than the target SINR (T). We have:

$$p_c(T, \lambda_n, \alpha) = \int_0^\infty P(\mathrm{SINR} \geq T) f_R(r)\mathrm{d}r \tag{4.18}$$

Using the distribution of $f_R(r)$ derived in (4.4), we get

$$\begin{aligned} p_c(T, \lambda_n, \alpha) &= \int_0^\infty P\left(\frac{gpR^{-\alpha}}{\sigma^2 + I_Z} \geq T\right) 2\pi\lambda_n r e^{-\pi\lambda_n r^2}\mathrm{d}r \\ &= \int_0^\infty P\left(g \geq \frac{T(\sigma^2 + I_Z)}{pR^{-\alpha}}\right) 2\pi\lambda_n r e^{-\pi\lambda_n r^2}\mathrm{d}r \end{aligned} \tag{4.19}$$

Since $g \sim \mathcal{E}(1)$, we get $p_c(T, \lambda_n, \alpha)$

$$\begin{aligned} &= \int_0^\infty 2\pi\lambda_n r e^{-\pi\lambda_n r^2 - Tp^{-1}r^\alpha\sigma^2} . E_{I_Z}\left[e^{-Tp^{-1}r^\alpha I_Z}\right] \mathrm{d}r \\ &= \int_0^\infty 2\pi\lambda_n r e^{-\pi\lambda_n r^2 - Tp^{-1}r^\alpha\sigma^2} . L_{I_Z}\left(Tp^{-1}r^\alpha\right) \mathrm{d}r \end{aligned} \tag{4.20}$$

where L_{I_Z} is the interference Laplace transform computed in (4.17). This gives a coverage expression

$$p_c(T, \lambda_n, \alpha) = 2\pi\lambda_n \int_0^\infty e^{-\pi\lambda_n r^2}\, e^{-Tp^{-1}r^\alpha\sigma^2} L_{I_Z}(Tp^{-1}r^\alpha) r\mathrm{d}r \tag{4.21}$$

From (4.17) we have

$$\begin{aligned} L_{I_Z}\left(Tp^{-1}r^\alpha\right) &= \exp\left(-2\pi\lambda_n \int_r^\infty \frac{1}{1+(Tr^\alpha)^{-1}x^\alpha} x\mathrm{d}x\right) \\ &= \exp\left(-2\pi\lambda_n \int_r^\infty \frac{T}{T+(x/r)^\alpha} x\mathrm{d}x\right) \end{aligned} \tag{4.22}$$

Change the variable $t = (x/r)^2 \rightarrow x\mathrm{d}x = \frac{r^2}{2}\mathrm{d}t$, we have:

$$\begin{aligned} L_{I_Z}\left(Tp^{-1}r^{\alpha}\right) &= \exp\left(\frac{-2\pi\lambda_n r^2}{2}\int_1^{\infty}\frac{T}{T+t^{\alpha/2}}\mathrm{d}t\right) \\ &= \exp\left(-\pi\lambda_n r^2\int_1^{\infty}\frac{T}{T+t^{\alpha/2}}\mathrm{d}t\right) \end{aligned} \tag{4.23}$$

Plugging (4.21) into (4.8), we have the successful access probability:

$$p_s = 2\pi\lambda_n e^{-\lambda_T/N_{ra}}\int_0^{\infty} e^{-\pi\lambda_n r^2}\, e^{-Tp^{-1}r^{\alpha}\sigma^2} L_{I_Z}(Tp^{-1}r^{\alpha})r\mathrm{d}r \tag{4.24}$$

The following equation provides the final expression of uplink coverage probability for the case assuming no noise ($\sigma^2 = 0$)

$$p_c(T,\lambda_n,\alpha) = \int_0^{\infty} 2\pi\lambda_n r e^{-\pi\lambda_n r^2} L_{I_Z}\left(Tp^{-1}r^{\alpha}\right)\mathrm{d}r \tag{4.25}$$

The successful access probability in the special case ($\sigma^2 = 0$):

$$\begin{aligned} p_s &= e^{-\lambda_T/N_{ra}}\int_0^{\infty} 2\pi\lambda_n r e^{-\pi\lambda_n r^2}\exp\left(-\pi\lambda_n r^2\int_1^{\infty}\frac{T}{T+t^{\alpha/2}}\mathrm{d}t\right)\mathrm{d}r \\ &= e^{-\lambda_T/N_{ra}}\int_0^{\infty} 2\pi\lambda_n r e^{-\pi\lambda_n r^2(1+L_t)}\mathrm{d}r \end{aligned} \tag{4.26}$$

where $L_t = \int_1^{\infty}\frac{T}{T+t^{\alpha/2}}\mathrm{d}t$.

4.3.3 Performance results

Evaluation Setup

In this section, we evaluate the performance of a NB-IoT uplink network and validate our analytical model for coverage probability derived in the previous section using Monte-Carlo simulations. The density of sensor devices $\lambda_n = 60680$ sensors/km^2 and the transmit power $p = 1W$.

We assume the number of RA preambles N_{ra} as 48 [36] and noise power σ^2 to be 0. We define the arrival rate as $\lambda = N_{dev}/I_{arr}.I_p$, where N_{dev} is the total number of devices, I_{arr} is the average packet inter-arrival time for a device (in Release 13, I_{arr} has been considered as about 10.8h), and I_p is the periodicity of the NPRACH (which has the value of 2.56 second).

Results

Figure 4.2 presents the coverage probability of the uplink NB-IoT as a function of the SINR threshold for the path loss exponent values of $\alpha = 2.5, 3, 4, 5$. We observe that the simulation model is close enough to the uplink analytical result derived in (4.25). Thus as α increases, the coverage probability increases due to the reduction in interference, especially from those devices located at the cell edge. We also note that for the very low SINR threshold $T < -15$ dB, the difference in coverage probability for $\alpha = 4, 5$, and 5 is negligible. As increases, the

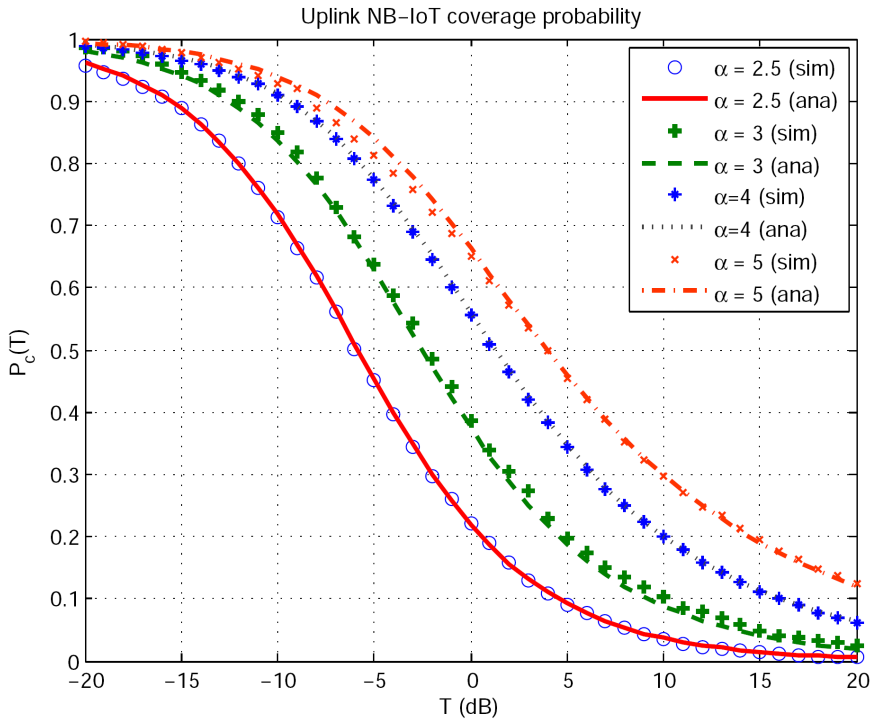


Figure 4.2: Uplink NB-IoT coverage probability.

coverage probability curves shift higher with = 4 providing much higher coverage probability than other cases, especially for SINR thresholds > 5 dB.

Figure 4.3 depicts the successful access probability according to the SINR threshold if we set the different number of repetitions. The successful access probability is improved as N_r increases due to the increase of the coverage probability. If we assume that the SINR threshold is -5dB, the successful access probability can be regarded as about 0.9 when $N_r = 4$.

Maximum Coupling Loss (MCL)

MCL [37] is a measure to describe the amount of coverage a system or design can support. Sometimes, the definition "km of coverage" is not an appropriate measure as it highly depends on the carrier frequency and the environment (e.g., indoor, outdoor, urban, sub-urban, and rural). Therefore, MCL is a better performance metric for the design because it is independent of frequency and environmental factors (LTE systems can operate at approximately 142 dB MCL). Table 4.1 shows the inputs and calculation for uplink MCL (from TR 36.888):

Required SNR

The target signal to noise ratio (SNR) in dB is computed base on a MCL. The relation between SNR and the target MCL is given by (4.27):

$$\mathrm{SNR}_{\mathrm{required}} = \mathrm{P}_{\mathrm{TX}} + 174 - \mathrm{NF} - 10\log_{10}(\mathrm{BW}) - \mathrm{MCL} \tag{4.27}$$

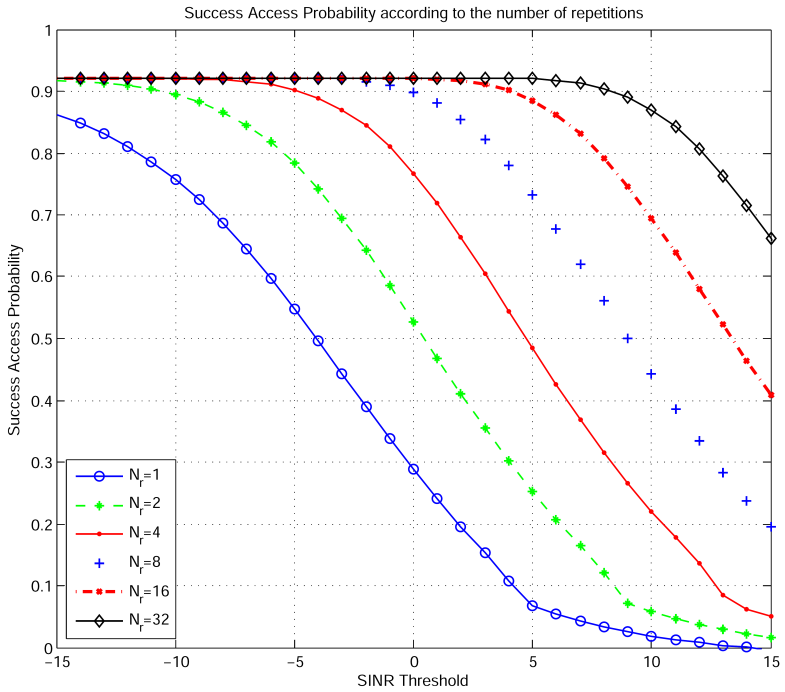


Figure 4.3: Successful access probability in uplink NB-IoT.

NB-IoT can operate at 164 dB MCL [38], which translates to a coverage enhancement of 20 dB. The required SNR varies from -11.8 dB to 14.3 dB depending on the number of tones and the subcarrier spacings.

Figure 4.4 presents the access success probability according to the distance from the tagged BS. Similar to the result in Figure 4.3, the access success probability is improved as N_r increases due to the increase of decoding success probability. In addition, the performance gets worse as N_{dev} increases due to the increase of collisions and outage probability. If we choose the threshold of the access success probability is 0.7, the cell radius can be regarded as about 12 km when $N_r = 128$ and $N_{dev} = 5 \times 10^4$.

Parameter	**Values**
(1) Transmitted Power P_{TX}	23 dBm - Uplink
(2) Thermal noise density	-174 dBm/Hz
(3) Receiver noise figure (NF)	3 dB
(4) Occupied channel bandwidth	$10\log_{10}$(BW)
(5) Effective noise power	(2) + (3) + (4) dBm
(6) Required SNR	-11.8 to 14.3 dB
(7) Receiver sensitivity	(5) + (6) dBm
(8) MCL	(1) - (7) dB

Table 4.1: Uplink NB-IoT parameters

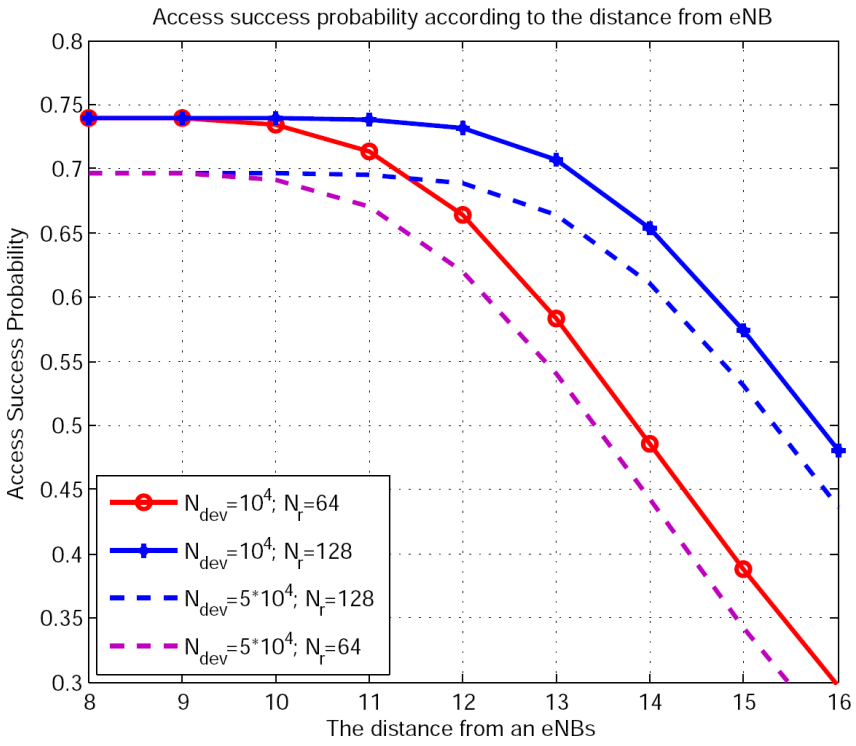


Figure 4.4: Success access probability according to the distance from eNB.

4.3.4 Conclusions

This section has presented expressions for the coverage probability and successful access probability in the NB-IoT uplink. The expressions are based on an analytical model utilizing stochastic geometry and are a function of system design parameters, including sensor device density and SINR targets. The simulations show that the proposed model is accurate. The next section will propose a new approach to predict the coverage probability in the multi-cell NB-IoT networks based on Machine Learning techniques.

4.4 Coverage analysis in a multi-cell NB-IoT network

In the previous section, we have derived the coverage probability in a single-cell NB-IoT network using stochastic geometry. We realized that the existing stochastic geometry-based analytical expressions for coverage are valid for a restricted set of network scenarios. Applying such a method for more realistic network scenarios such as multi-cell NB-IoT has so far proven impracticable. In this section, we present a new approach that can predict the coverage probability in a more complex network scenario - a multi-cell NB-IoT network. We design a coverage probability predictor based on Deep Neural Networks (DNNs). The presented analysis shows that our proposed model achieves an average of 97 % of accuracy on the considered dataset and also provides a comparison between other supervised algorithms. The methodology and the achieved results are novel in the context of the NB-IoT network. In

summary, the contributions are as follow:

- The coverage probability of NB-IoT network as a form of the sigmoid function;
- Estimate the coverage probability using Deep Neural Networks: we design a DNN network, and we tune the training parameters to obtain the maximum accuracy. A comparison with other classes of algorithms demonstrates the advantages of our approach.

4.4.1 Model Architecture

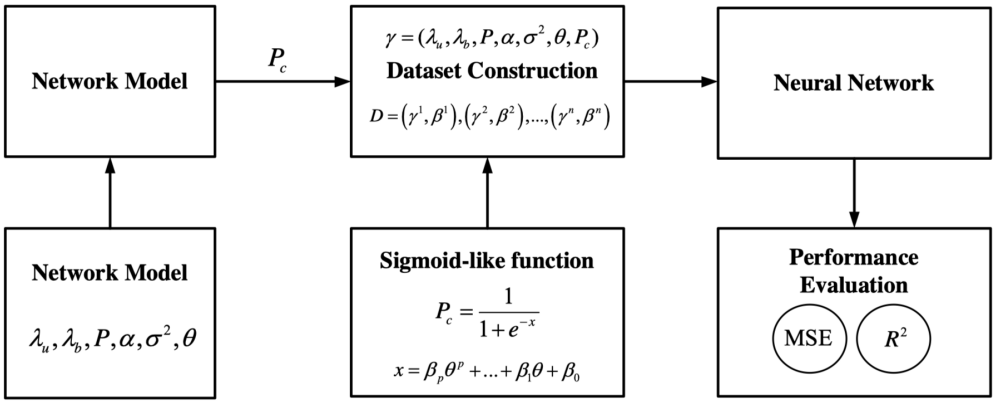


Figure 4.5: Proposed architecture for the coverage probability prediction.

The proposed architecture for the coverage probability prediction is depicted in Figure 4.5. The design of the prediction system includes a Neural Network unit. In our case, we state the problem as a supervised multivariate prediction of the parameterized sigmoid-like function that describes the coverage probability of a typical user in an NB-IoT network. The objective is to minimize the prediction error given the generated information. First, based on the proposed network model, we run a large number of simulations to get the dataset of coverage probability with different network parameters, namely: UE intensity λ_u, eNB intensity λ_b, UE transmit power P, path loss exponent α, noise power σ^2. Then, we show that the coverage probability can be closely approximated by using a parameterized sigmoid-like function $\boldsymbol{\beta}$. Last, the neural network is used to estimate the parameters of the sigmoid-like function from the input features that characterizes the network parameters. The performances of model are evaluated by the mean square error and the coefficient of determination metrics.

Further detail of each step will be introduced below.

4.4.2 System model and assumptions

Figure 4.6 gives a visual representation of the uplink system model and the relationship between system parameters. We consider a noise limited system where the eNB are randomly distributed on the 2D plane following a homogeneous Poisson point process (PPP) Φ_b with density λ_b and the NB-IoT devices are assumed to form a realization of a homogeneous PPP Φ_u with density λ_u. The spatial PPP corresponds to a uniform distribution of user devices in the network, which is the baseline assumption for many cellular system studies [39]. We

assume that each NB-IoT device is served by the eNB, which provides the best signal-to-interference-and-noise ratio (SINR).

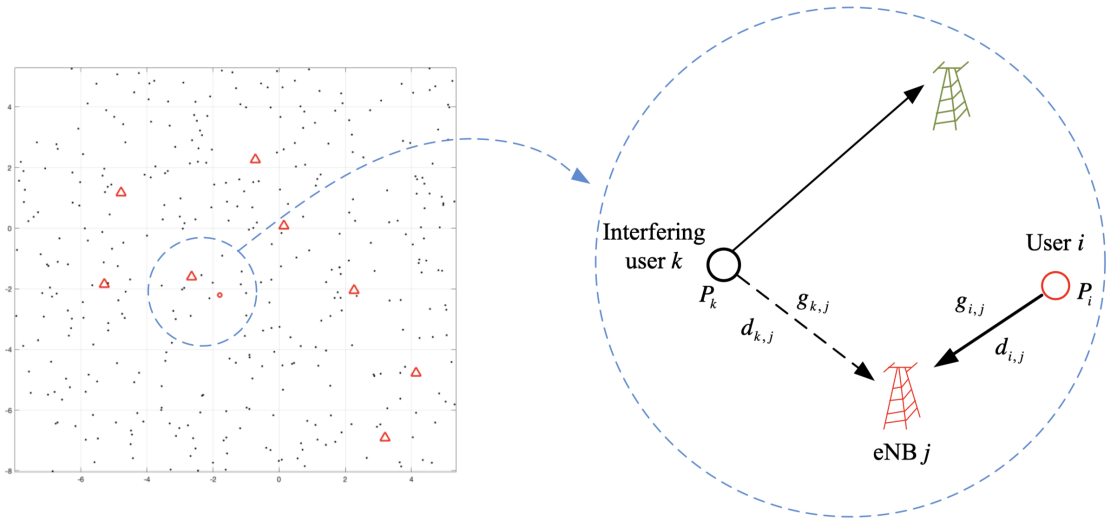


Figure 4.6: Representation of the uplink system model.

To model the channel, the path-loss is assumed to be inversely proportional to distance with path loss exponent given by α. We consider small-scale Rayleigh fading between the devices and the eNB under consideration. Thus the received power of the desired signal at the serving eNB j is given by $d_{i,j}^{-\alpha} g_{i,j} P_i$, where $g_{i,j}$ denotes the small-scale fading power gain, $d_{i,j}$ is the distance between the device i and eNB j, P_i is the transmit power of NB-IoT device i. The noise power is assumed to be σ^2.

The interference at a randomly chosen eNB is the sum of powers from all transmitting devices except the tagged device i. To model uplink interference, we denote the set of interfering devices by $\{Z\}, \{Z\} = \Phi_u \setminus i$, the distance of an interfering device $k \in Z$ to the eNB j by $d_{k,j}$ and the small-scale Rayleigh fading between the interfering devices and the eNB by $g_{k,j}$. We assume that each interfering device transmits a power P_k.

Under this system model, the associated SINR experienced by the selected NB-IoT device i connected to the eNB j is given by

$$\text{SINR}_{i,j} = \frac{d_{i,j}^{-\alpha} g_{i,j} P_i}{\sigma^2 + I_Z} \tag{4.28}$$

where for an interfering set of devices $\{Z\}$

$$I_Z = \sum_{k \neq i, k \in \Phi_u} d_{k,j}^{-\alpha} g_{k,j} P_k \tag{4.29}$$

In the proposed scenario, all NB-IoT devices are assumed to transmit an equal transmit power, i.e. $P_i = P_k = P, \forall i, j$; the small-scale Rayleigh fading $g_{i,j}$ and $g_{k,j}$ are assumed to follow an exponential distribution with unit mean $E(1)$.

4.4.3 The coverage probability

The uplink coverage probability is defined as the complementary cumulative distribution function (ccdf) of the uplink SINR as

$$P_c(\theta) = \mathrm{P}\left(\mathrm{SINR} > \theta\right) \tag{4.30}$$

which is the probability that the uplink SINR at randomly chosen eNB is greater than the SINR threshold θ. It can also be visualized as being the average area or the average fraction of devices in coverage.

We have generated a large number of network realizations to get the curves of the average coverage probability with different network parameters, such as: path-loss exponent α, eNB intensity λ_b, UE intensity λ_u, transmit power P, noise power σ^2 and radius of area R. The examples of these coverage probability curves are provided in Figure 4.7.

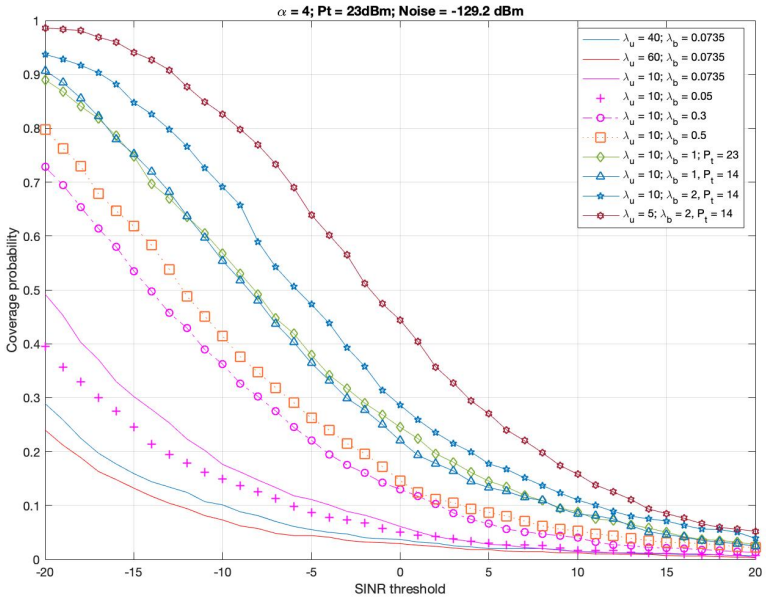


Figure 4.7: The coverage probability curves.

In addition, we realize that the average coverage probability curves can be modeled by using the curve-fitting models. One of the most effective methods of curve-fitting is the method of least squares. The method of least squares assumes that the best fit curve of a given type is the curve that has the minimum sum of deviation, i.e., least-square error from a given set of data. Suppose that the set of data points is $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$. According to the method of least square, the best fitting curve has the property that $\sum_1^n e_i^2 = \sum_1^n [y_i - f(x_i)]^2$ is minimum, where $f(x)$ is the fitting function and e_i is the deviation from each data point i,i.e., $e_i = y_i - f(x_i)$. In this work, the parameterized sigmoid-like function $f(x) = \frac{1}{1+e^{-x}}$ is proposed to describe the coverage probability in a random NB-IoT network, where x is the function of curve-fitting parameter $\boldsymbol{\beta} = (\beta_p, ..., \beta_1, \beta_0)$ and the SINR threshold θ. Thus, the

coverage probability function is defined as:

$$P_c(\theta) \approx \frac{1}{1+\exp\left(-\beta_p\theta^p - ... - \beta_1\theta - \beta_0\right)} \tag{4.31}$$

where p is the order of the polynomial in the sigmoid-like function. Figure 4.8 provides an example of curve-fitting using the sigmoid-like approximation. In our case, $p = 2$ is sufficient for providing a good fit of coverage probability curves.

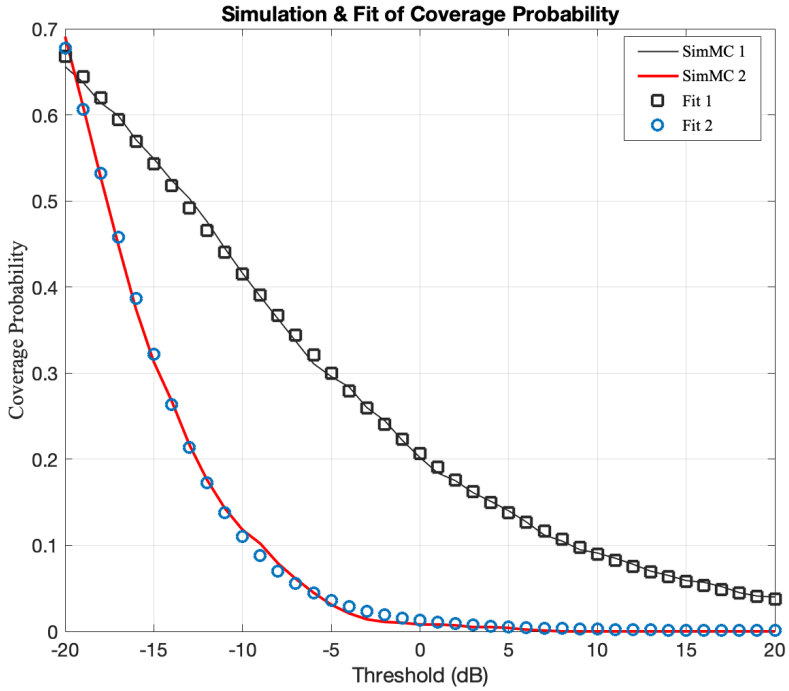


Figure 4.8: An example of curve-fitting using the sigmoid-like approximation.

Finally, the coverage probability can be determined via the parameter vector $\boldsymbol{\beta}$. In order to estimate values of $\boldsymbol{\beta}$ without running Monte-Carlo simulations, we use machine learning techniques to learn the relationship between the network parameters and the sigmoid-like function parameter vector $\boldsymbol{\beta}$.

4.4.4 Data Preparation

The input of Neural Network is a dataset, which is generated by running a large number of simulations with various network features. We denote the network parameters by vector $\boldsymbol{\gamma} = (\alpha, \lambda_b, \lambda_u, P, \sigma^2, R, \theta)$. The input features of the neural network are the network parameters, which are those that the network takes on the input layer in order to make a prediction. The output features are the elements of parameter vector $\boldsymbol{\beta}$, which are collected by fitting the model described in equation 4.31. The generated dataset, denoted by D, consists of

$$\mathrm{D} = (\boldsymbol{\gamma}^1, \boldsymbol{\beta}^1), (\boldsymbol{\gamma}^2, \boldsymbol{\beta}^2), ..., (\boldsymbol{\gamma}^n, \boldsymbol{\beta}^n) \tag{4.32}$$

where ($\boldsymbol{\gamma}^j$ and $\boldsymbol{\beta}^j$) denote the j-input and j-output feature vectors, n is the volume of the dataset, which equal the number of network realizations. The construction of the dataset is presented in Table 4.2.

Input feature vector						Output feature vector		
α	λ_u	λ_b	P	σ^2	R	β_2	β_1	β_0
4	15	0.70695	23	-129.2	12	0.0010297	-0.13669	-1.7599
6	11	0.94622	22	-129.2	12	0.00029088	-0.10033	-0.71678
3	16	0.48538	23	-129.2	12	0.002205	-0.13396	-2.8147
...	...	...	...	...	...	...	...	...

Table 4.2: Dataset construction

As can be seen, the variables in the input vector have different units. That may increase the difficulty of the learning process and result in large error gradient values. Therefore, scaling input and output variables is a critical step in using neural network models.

The input variables should be small values, probably in the range of $(0-1)$ or standardized with a zero mean and a standard deviation of one. There are two types of scaling of data, namely normalization, and standardization. In this work, the input variables are standardized by removing the mean and by scaling to unit variance.

The dataset is then randomly split into two sets: a training set and the test set, which represent 80% and 20% of the entire dataset, respectively.

4.4.5 Neural network

The general introduction of the Neural network has been presented in Chapter 3. In mathematical terms, the weights of the deep neural networks (DNNs) can be learned by minimizing a loss function using the gradient descent method through backward propagation, following the fundamental chain rule. We illustrate the principles of the forward propagation (learning) and the backward propagation (interference) processes of a deep neural network in Figure 4.9.

Forward Propagation

Suppose our dataset consists of m input-output pairs $(x^{(i)}, y^{(i)}), i = 1, ..., m$. The DNN takes a vector of input $x^{(i)} = [x_1, x_2, ..., x_n]$ and produces an output $\tilde{y}^{(i)}$. Figure 4.9 shows the DNN architecture with 5 layers,i.e., an input layer, 3 hidden layers, and an output layer. The operation of each layer can be denoted with a matrix of weight $W^{(l)}$, where l refers to the layer number (we consider $l = 0$ for the input layer). The dimensions of this matrix are equal to the number of inputs multiplied by the number of hidden neurons. The output of each hidden layer is evaluated as follows:

$$z^{(l)} = W^{(l)} x^{(i)} + b^{(l)} \tag{4.33}$$

$$a^{(l)} = \sigma\left(z^{(l)}\right) \tag{4.34}$$

where $z^{(l)}$ is the linear combination and $\sigma()$ denotes the activation function, aiming at improving the non-linearity and representability of the model. The output $a^{(l)}$ is subsequently

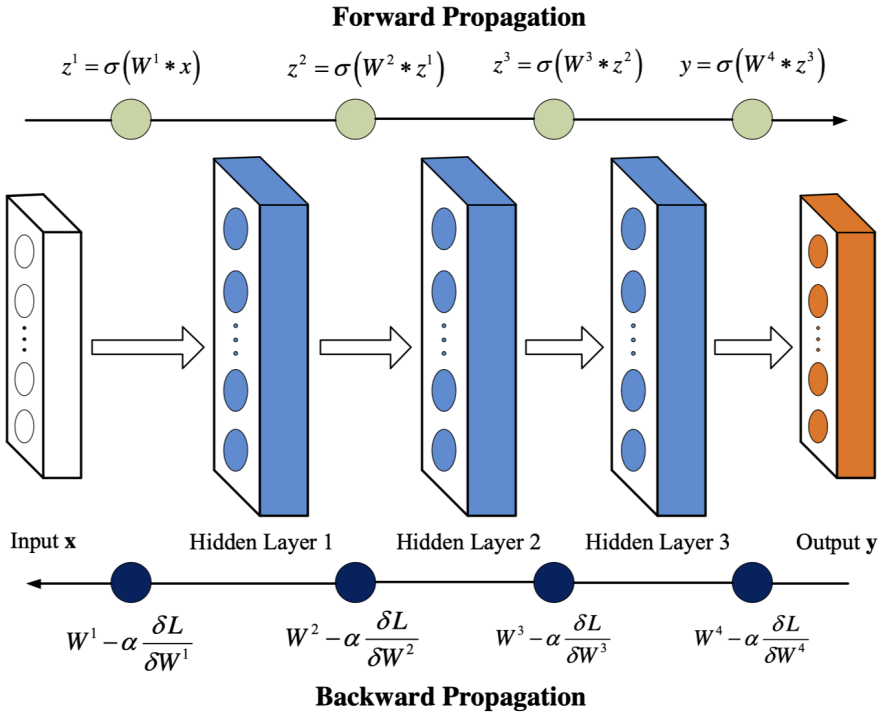


Figure 4.9: Deep neural network architecture.

provided as input to and processed by the following layer, which eventually produces a final output of $\tilde{y}$:

$$z^{(1)} = W^{(1)}x^{(i)} + b^{(1)} \tag{4.35a}$$
$$a^{(1)} = \sigma\left(z^{(1)}\right) \tag{4.35b}$$
$$z^{(2)} = W^{(2)}a^{(1)} + b^{(2)} \tag{4.35c}$$
$$a^{(2)} = \sigma\left(z^{(2)}\right) \tag{4.35d}$$
$$z^{(3)} = W^{(3)}a^{(2)} + b^{(3)} \tag{4.35e}$$
$$_i = a^{(3)} = \sigma\left(z^{(3)}\right) \tag{4.35f}$$

In the case of a regression problem, we can use tanh, sigmoid, or ReLu functions as the activation functions.

The purpose of training is to optimize the weights and bias terms in order to minimize the loss function. This can be achieved by the backward propagation through gradient descent.

Backward Propagation with Gradient Descent

In this section, we describe the loss and cost functions and the backward propagation algorithm with gradient descent. Loss function $L\left(\tilde{y}^{(i)}, y^{(i)}\right)$ is defined as a function that models the error between the actual output $\tilde{y}^{(i)}$ and the desired (real) output $y^{(i)}$. A natural and common choice

for the loss function is Mean Square Error (MSE), defined as the average squared difference between the estimated values and true values.

$$L\left(\tilde{y}^{(i)}, y^{(i)}\right) = \text{MSE}(\tilde{y}^{(i)}, y^{(i)}) = \sum_{i=1}^{m} \left(y^{(i)} - \tilde{y}^{(i)}\right)^2 \tag{4.36}$$

Besides the advantage of simple computation and being applicable for almost scenario, the MSE in some cases can slow down the learning algorithm. In the binary classification problem, where the desired and actual output data belong to the interval $[0, 1]$, the learning algorithm is typically observed by using the cross-entropy loss function $H(\tilde{y}^{(i)}, y^{(i)})$:

$$L\left(\tilde{y}^{(i)}, y^{(i)}\right) = H(\tilde{y}^{(i)}, y^{(i)}) = -\sum_{i=1}^{m} \left(y^{(i)} \log(\tilde{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \tilde{y}^{(i)})\right) \tag{4.37}$$

The goal of the training algorithm is to optimize the DNN weights and bias terms in order to minimize the loss between the actual and the desired outputs. Given m samples of input data, we define the *cost function* $J(W, b)$ as the average of the m losses calculated over the m training samples:

$$J(W, b) = \frac{1}{m} \sum_{i=1}^{m} L^{(i)} \tag{4.38}$$

Thus, the backpropagation procedure can be stated in Algorithm 1:

Algorithm 1 Backpropagation Algorithm with Gradient Descent

1: **for** $i = 1 \rightarrow m$ **do**
2: Training input $x^{(i)}$ with desired output $y^{(i)}$;
3: **Forward Propagation:** Compute the actual output $^{(i)}$;
4: **Backward Propagation:** Compute the gradient $\frac{\delta L}{\delta W}$;
5: **end for**
6: $J = \frac{1}{m} \sum_{i=1}^{m} L^{(i)}$;

Stochastic Gradient Descent

In the deterministic gradient descent approach, one must forward-propagate and backward-propagate all m samples of the training set. This can be computationally challenging for extremely large datasets. The complexity is proportional to m when computing the gradient of the loss function. To address this issue, state-of-the-art training algorithms for DNNs employ a variant of the gradient descent algorithm known as Stochastic Gradient Descent (SGD) [40]. The term "stochastic" hints already the use of particular approximation techniques which involve randomness (stochastic approximations). In SGD, we approximate the gradient $\frac{\delta L}{\delta W}$ by random select component $\frac{\delta L^{(i)}}{\delta W}$, where i being chosen randomly out of $(1, ..., m)$. In another word, for any single layer l, we can update all parameters $(W^{(l)}, b^{(l)})$ as follow:

$$W^{(l)} = W^{(l)} - \alpha \frac{\delta L}{\delta W^{(l)}} \tag{4.39}$$

$$b^{(l)} = b^{(l)} - \alpha \frac{\delta L}{\delta b^{(l)}} \tag{4.40}$$

where α is the learning rate. It controls how fast the algorithm reduces the cost function, and thus learns. It is important to repeat the random selection of the index i during each new iteration. While traditional gradient descent algorithms can use a fixed α and converge as long as α is not too large, the SGD uses a variable α to be used in iteration k, $\alpha = 1/k$. The step size needs to be gradually decreased while running SGD update to avoid the accumulation of the gradient noise ϵ.

In practice, research and applications use mini-batch gradient descent. This is a compromise between gradient descent and stochastic gradient descent. More precisely, if we denote the number of examples in the mini-batch by B, the cost function J_{mb} is defined as follows:

$$J_{mb} = \frac{1}{B} \sum_{i}^{B} L^{(i)} \tag{4.41}$$

Each time a gradient descent step is taken, the estimated gradient is evaluated based on a new, randomly selected mini-set. The overall procedure is provided in Algorithm 2:

Algorithm 2 Stochastic Gradient Descent with Mini-batch

1: Set $\epsilon > 0, W, b$
2: **while** Validation Error larger than ϵ **do**
3: Sample a random mini-batch;
4: Compute gradient by Algorithm 1;
5: $W = W - \alpha J_{mb}$;
6: $b = b - \alpha J_{mb}$;
7: **end while**

In addition, the goal of deep learning is not only to minimize the cost function but also to ensure a low generalization gap. Besides the objective of achieving a low training error, there is an equally important task is that of tuning the network hyperparameters (e.g., the number of layers L, the number of neuron per layer N_l, the size of training set m, etc.) to fit the training data, avoiding both underfitting and overfitting.

4.4.6 Performance Results

The objective of our proposed model is to find the optimum value of $\boldsymbol{\beta}$ for which we obtain the lowest error. We also compare our solution with other Machine learning algorithms. Based on the Monte-Carlo simulations, the dataset is constructed using Matlab and Simulink curve-fitting tools. The performance tests have been carried out using on cloud environment using Google Colaboratory. There are 2000 samples in our dataset. We divide the dataset into training and test sets with a split ratio of 80% and 20%. These sets are balanced, as they contain random network features. The DNN algorithm has been implemented in Python. We have used the popular *sklearn* library for the implementation of deep Neural Networks.

Figure 4.10 represents the graphical representation of our proposed neural network model with the weights on each connection. The input features include the network parameters $\boldsymbol{\beta} = (\alpha, \lambda_u, \lambda_b, P_t)$ and the output is the parameters of sigmoid-like function $\boldsymbol{\beta} = (\beta_2, \beta_1, \beta_0)$. The black lines show the connection between each layer and the weights on each connection, while the blue lines show the bias term added in each step.

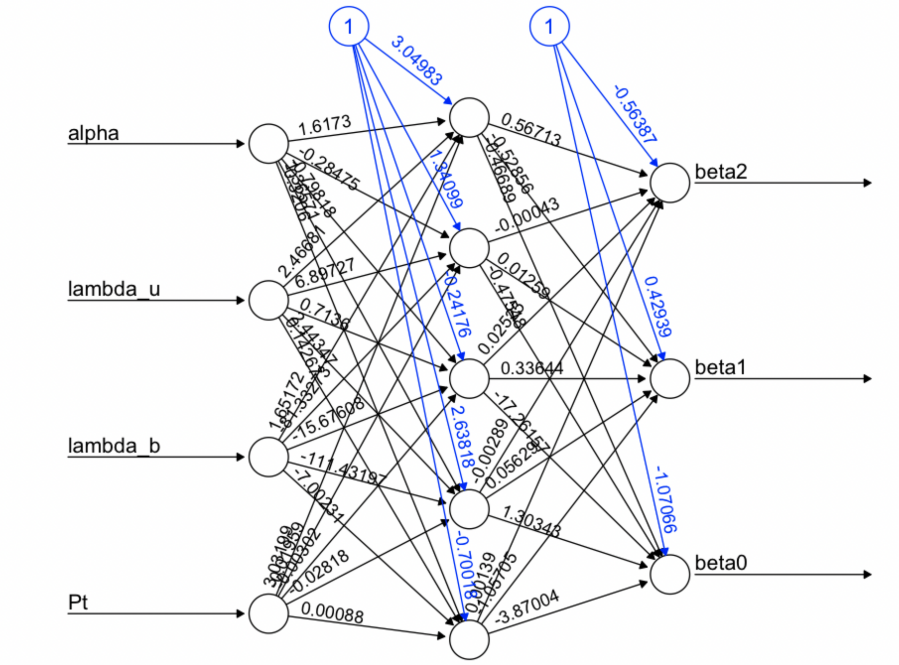


Figure 4.10: The architecture of applied neural network.

Performance Evaluation

The performances of the proposed model are measured with two efficiency terms as follows.

- *The coefficient of determination* (R^2 value) is the proportion of the variance in the dependent variable that is predictable from the independent variables. The best possible score of R^2 is 1.0, when the modeled values exactly match the observed values. Assume a dataset has n values $(y_1, ..., y_n)$ and each associated with a predicted value $(\hat{y}_1, ..., \hat{y}_n)$. The most general definition of R^2 is

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \tag{4.42}$$

where SS_{tot} is the total sum of square

$$SS_{tot} = \sum_i (y_i - \bar{y})^2,$$

$\bar{y}$ is the mean of the observed data

$$\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$$

and SS_{res} is the residual sum of squares

$$SS_{res} = \sum_i (y_i - \hat{y}_i)^2.$$

The number of hidden layer	MSE	R^2
1	0.0336	0.9576
2	0.0201	0.9746
3	0.0201	0.9745
4	0.0185	0.9767
5	0.0215	0.9728
10	0.0234	0.9704
20	0.0336	0.9574
30	0.0321	0.9594

Table 4.3: Performance efficiencies of different network architectures.

- *Mean Squared Error* (MSE) is the squared difference between the predicted values and the actual value. The MSE is always non-negative, and the values closer to zero are better. The definition of an MSE is

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2) \tag{4.43}$$

The performance efficiencies of difference MLP architectures are presented in Table 4.3. Comparing the models that used different hidden layers, we observed that four hidden layers of MLP are sufficient to achieve very high accuracy, and having more hidden layers does not significantly improve the prediction accuracy.

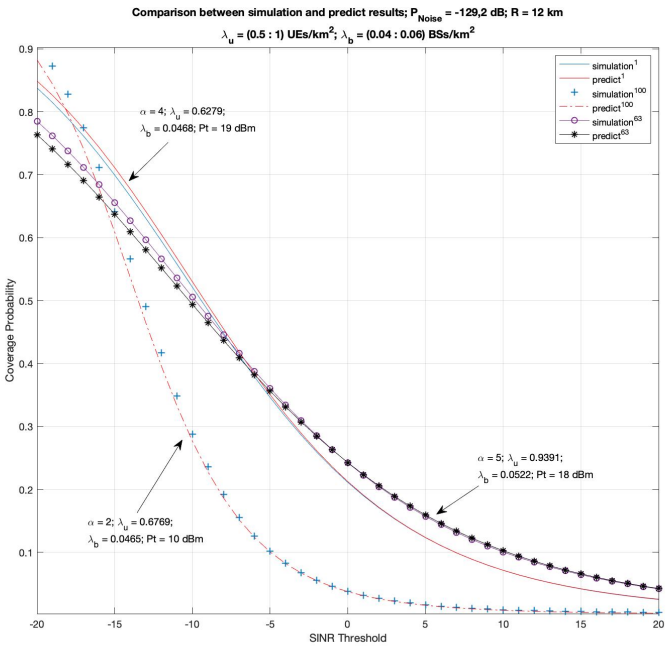


Figure 4.11: Predicted coverage and Monte-Carlo simulations.

Algorithm	Parameters	Reference
K-Nearest Neighbours	$K = 20$ $p = 2$ *metric*=Minkowski	K: number of neighobours for queries p: distance metric parameter $p = 2$ amounts to using the Eclidean distance
Linear Regression	default	Uses the default values of all parameters
SVM + MO Regressor	$C = 100$ kernel='linear'	C: penalty parameter for the error term extended to multi-output regressor

Table 4.4: Configuration parameters of different algorithms

Algorithms	MSE	R^2
MLP	0.0190	0.9760
K-Nearest Neighbours	0.1135	0.8567
Linear Regression	0.1333	0.8317
SVM combined MO regressor	0.1565	0.8023

Table 4.5: Accuracy of predictions of the different algorithms.

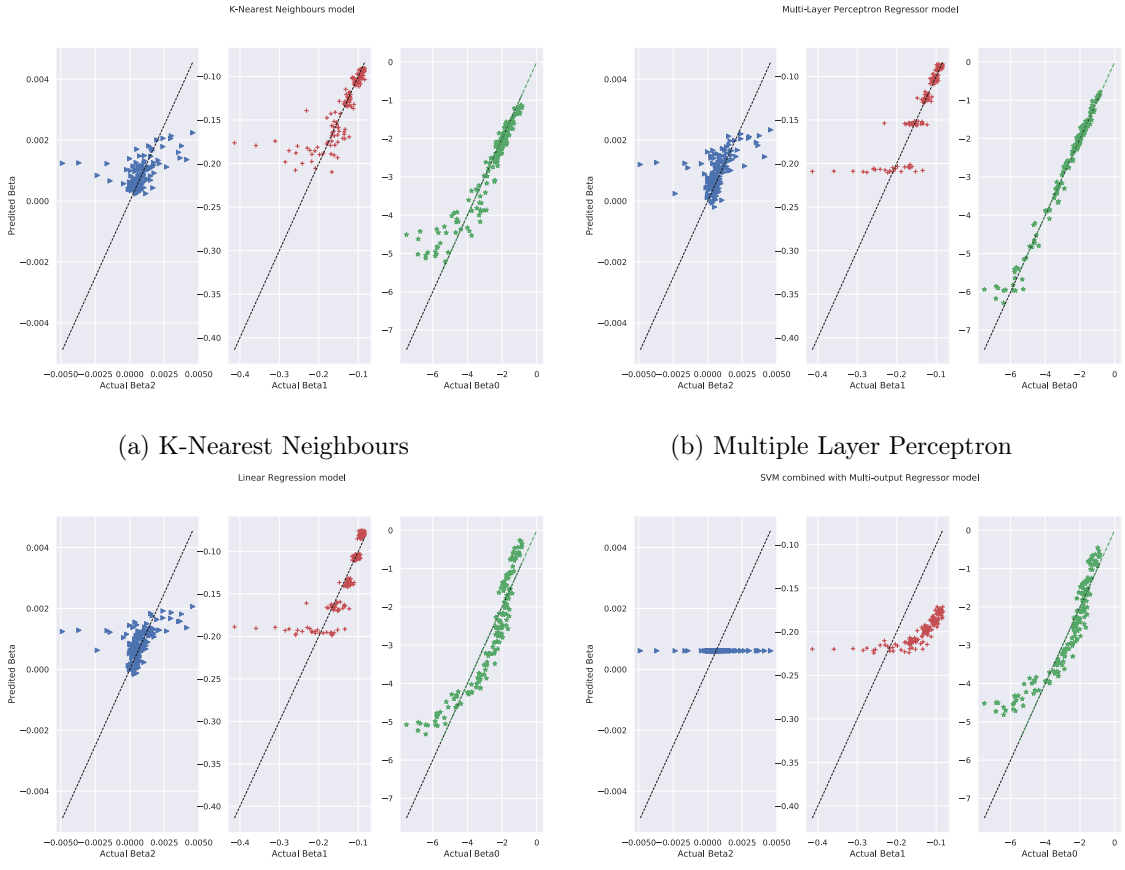
Figure 4.11 shows the coverage probability based on Monte-Carlo simulations and the NN-based coverage probability prediction. There are three random network scenarios, which are represented by different network parameters. As shown in the figure, the NN-based model provides a very accurate prediction of the coverage probability.

Comparison with different Machine learning methods

Other standard regression techniques have been tailored to the $\boldsymbol{\beta}$ prediction task. The selected algorithms are K-Nearest Neighbours, Linear Regression, and Support Vector Machine (SVM) Regressor combined with Multi-output (MO) Regressor. The detail of each algorithm is introduced in Chapter 3. Configuration parameters of these algorithms are provided in Table 4.4. MSE and R^2 values are used to analyze model accuracy for each selected algorithm.

The predictive accuracy of these above algorithms is evaluated by the coefficient determination R^2 and Mean Square Error (MSE). The predictive estimation results are summarized in Table 4.5. The obtained MSE and R^2 values of MLP algorithm are 0.019 and 0.9760, respectively. These metrics signify the MLP algorithm outperforms the other three ML algorithms. The MSE and R^2 values of K-Nearest Neighbours are 0.1135 and 0.8657, comparatively better than Linear Regression and SVM combined with Multi-output Regressor algorithms.

In addition, the performance comparison of the four algorithms is aslo illustrated by scattering plots in Figure 4.12. A scatter plot shows the relationship between two variables (actual and predicted) as dots in two dimensions, one axis for each attribute. Each pair of attributes in beta is illustrated in the plot. By visually examining the plot, we can realize that the predictions made by the Multiple Layer Perceptron are more concentrated around the line than those made by the other algorithms. Note that a perfect alignment with the line indicates a MSE of 0 and thus is an ideal perfect prediction.



(a) K-Nearest Neighbours

(b) Multiple Layer Perceptron

(c) Linear Regression

(d) SVM combined with Multi-Output Regressor

Figure 4.12: Scattering graphs of predicted against actual values in different models

4.4.7 Conclusion

The proposed study clearly shows that using Machine Learning techniques gives a new approach for the prediction of coverage probability in NB-IoT networks. By using a neural network, it will be possible to predict the coverage probability that aims at optimizing the NB-IoT networks based on the input parameters of networks. The results show that MLP model has a better performance compared to other supervised learning algorithms in terms of mean squared error and the coefficient of determination metrics.

4.5 Summary

In this chapter, the uplink NB-IoT network performance is analyzed using stochastic geometry and machine learning approaches. In section 4.3, user equipment association based on the nearest distance with eNB is employed. The expression for the coverage probability and the success access probability is derived. The coverage probability is defined as the probability that a given device can achieve some thresholds SINR. The coverage probability can be viewed as complementary of the outage probability. It is shown that the implementation of a repetition transmission scheme can considerably improve the access success probability.

In section 4.4, the coverage probability in a multicell NB-IoT network is predicted using machine learning. By running a large number of simulations, we have a group of coverage probability with different network parameters. Each coverage probability of a network realization can be approximated by the parameters of sigmoid-like functions. The neural network is proposed to estimate these parameters from the feature set that characterizes the random NB-IoT network. In this way, the performance in the random NB-IoT networks can be analyzed more tractably and accurately.

4.6 Resulting research contribution

The research contribution resulting from the work done in this chapter is below:

- T. A. Nguyen, P. Martins, V. T. Nguyen and T. M. T. Nguyen, "A New Analytical Model for the Performance Evaluation of the Uplink Transmission in NB-IoT Networks," 2018 IEEE 88th Vehicular Technology Conference (VTC-Fall), Chicago, IL, USA, 2018, pp. 1-5.

  DOI: 10.1109/VTCFall.2018.8690677.

Chapter 5

NB-IoT network resource outage and Dimensioning

Radio resource outage is the main issue in designing of a network and, more particularly, in Narrowband Internet of Things (NB-IoT). Radio Resource Management (RRM) is used to optimize network setting parameters to serve all user devices when minimizing the probability of the radio resource outage event that arises when the management entity has no more radio resources to attribute the user devices. A network outage happens when the number of available resource blocks (RBs) in the network is smaller than the one required by the users, depending on the SINR value of user positions and the fading model. This Chapter focuses on the dimensioning in the downlink of a multi-cell NB-IoT network. We use stochastic geometry tools and the properties of Bell Polynomials to provide an analytical model to estimate the required number of resource blocks to avoid the radio resource outage. To this intent, we first derive the expression of the outage probability considering network parameters. Then, based on a given outage probability, we determine the number of RBs required in the NB-IoT network to guarantee the target outage probability, taking into account the repeated transmission feature of NB-IoT.

5.1 Introduction

The Internet of Things (IoT) has been in popularity in recent years. The number of IoT devices is increasing dramatically both in the number and the area of applications (health-care, logistic, agriculture, transportation, etc.). IoT devices have strong power consumption, deep coverage, and low device complexity requirement. Besides, the number of devices in IoT networks which is several thousand times greater than in Long Term Evolution (LTE) networks, is a challenge in the network design. NB-IoT is a Low Power Wide Area (LPWA) technology, which was standardized in the Third Generation Partnership Project (3GPP) release, specifies a new random access (RA) procedure and a new transmission scheme for IoT [9]. The advantages of NB-IoT network are providing deep coverage, low power consumption, and support of a large number of connections. NB-IoT can be deployed in any of the 2G/3G/4G spectrum (from 450 MHz to 3-5 GHz), since it achieves an excellent co-existence and compatibility performance with legacy cellular systems. NB-IoT needs only a small portion of the existing available cellular spectrum to operate without interfering with it. Hence, NB-IoT provides more reliability and more Quality of Serice (QoS) as it operates in a regulated

spectrum. Moreover, NB-IoT uses existing cellular network infrastructure, which reduces the deployment costs. Moreover, repeating the transmission of data and the associated control signaling several times has been utilized as a base solution to achieve coverage enhancement in NB-IoT. It is crucial to design an appropriate resource allocation scheme integrated with considering repetition factors for NB-IoT systems.

In our work, we focus on investigating the Radio Resource Management in the downlink of multi-cell NB-IoT.

One of the main objectives of Radio Resource Allocation in such a system is to determine the radio resources in the network to guarantee the required Quality of Service (QoS), IoT device density, and eNodeB (eNB) density. Resource allocation in the NB-IoT network is based on the resource block. The number of resource blocks allocated to a sensor node can be calculated from the channel quality and the data rate required by the application. If the total number of resource blocks required is greater than the cell capacity, packets will be lost.

The problem addressed here can then be stated as follows: how many radio resources must be assigned to an eNB to ensure a small outage probability? Given the number of the available resource block, what is the maximum load in terms of the number of user devices in a cell that can be tolerated? Both questions rely on accurate estimations of the outage probability. This work proposes an analytical model that calculates the required number of resource blocks as a function of the network parameters in multi-cell downlink NB-IoT. In this network, we consider that the distribution of users and eNBs follows a random Poisson Point Process (PPP).

5.2 Related Works

Poisson Point Process has been widely used in the literature [41] - [42] to model the positions of eNBs and sensor users in the cellular network. The problem of Orthogonal Frequency-Division Multiple Access (OFDMA) based on the network planning and dimensioning has been more recently under investigation. In [43], the authors propose a dimensioning of OFDMA systems focusing on link outage but not on the other parameters of the system. In [44], the authors give a general methodology for the dimensioning of OFDMA systems, which mixes a simulation-based determination to distribute the signal-to-interference-plus-noise ratio (SINR) and a Markov chain analysis of the traffic. In [45], [46], the authors propose a dimensioning method for OFDMA systems using Erlang's loss model and Kaufman Roberts recursion algorithm. In [47], the authors study the effect of Rayleigh fading on the performance of OFDMA networks. In [48], the authors give robust methods to compute an upper bound on the resource outage probability for LTE network.

However, few efforts have been done for the radio resource allocation specific to NB-IoT systems so far. Yu *et al.* [49] present a single-tone scheduling mechanism with the repetition number determination for NB-IoT uplink users. The key technologies of uplink scheduling, such as power control and transmission gap, are also analyzed. It has been shown that the proposed scheme reduces the active time and resource consumption but it is only valid for users with good channel conditions and large packet sizes. However, the proposed scheme has not considered the effect of intercell interference (ICI), which is one of the key performance degradation factors in NB-IoT. Similarly, in [50], performance analysis of resource unit configurations for uplink transmission in the NB-IoT is presented. The best resource configurations to be used by NB-IoT are discussed. However, they have not considered the

effect of repetition factor and ICI on the performance. Furthermore, an efficient small data transmission scheme for NB-IoT to reduce the impact of repetition based on control plane solution has been presented in [51]. The results have shown that the proposed scheme can increase the maximum number of served devices by about 60% compared with the conventional scheme. In the proposed scheme, the NB-IoT resource control (RRC) connection setup process is omitted for higher system capacity. If there is a downlink packet for a short time after transmitting an uplink packet, it may lead to an issue since a device can receive the downlink packet through the paging process.

In [52], the authors propose an uplink NB-IoT dimensioning model in which the interference is considered. The active sensors and collectors are randomly distributed according to a random Poisson Point Process. However, the proposed model is performed only on a single-cell with a single class of users. A theoretical framework for the upper bound on the achievable data rate in the presence of the control channel and repetition factor is presented in [4]. In addition, the authors propose an interference-aware resource allocation for NB-IoT by formulating the rate maximization problem. However, the proposed framework has not considered the distribution of users, which is the main factor of the dimensioning problems.

Compared to the existing works, the main contribution of our work is the derivation of a numerical model to calculate the required radio resource blocks in the multi-cell downlink NB-IoT network using the Bell polynomials. This model is very necessary for operators because it shows how to manage the available spectrum resource. Additionally, we show that the total number of the requested PRBs follows a compound Poisson distribution, and we derive the explicit expression of the outage probability as a function of different system parameters. This metric is defined as the probability that the requested resources exceed the available ones. It is often considered a primary standard for operators when it comes to resource dimensioning since it is related to the guaranteed quality of service.

This Chapter is structured as follows. We first introduce the system model of the NB-IoT system and set up the problem of radio resource allocation in Section 5.3. In Section 5.4, we consider the outage probability and derive the Bell polynomials method to find the number of resource blocks necessary to achieve a target outage probability. We then present numerical results to estimate the outage probability and analyze the relationship between the required number of RBs and the IoT device density in Section 5.5. Finally, conclusions are drawn in Section 5.6.

5.3 System model

In this section, we consider the performance of the downlink channel in the NB-IoT network. Networks are deployed according to a Poisson point process. Since we study an NB-IoT system, the downlink channel of each antenna is divided into subcarriers of 15 kHz. Assume that each cell uses a different frequency band from its neighboring cells and has M available resource blocks. These resource blocks are then allocated to the active users, whose positions are drawn according to a Poisson point process.

5.3.1 Sensor users model

We suppose that the positions of sensor users in the cell follow a spatial PPP Φ_u with intensity λ_u. In a given coverage area, the number of sensors follows a Poisson distribution while their locations follow a uniform distribution. From now on, the two terms of intensity and density

are used interchangeably. Let Φ_b be a point process of intensity λ_b, that represents the set of eNBs. Therefore, the average number of sensors in a cell inside the coverage area, denoted by u, can be calculated by

$$u = \lambda_u S_{area} \tag{5.1}$$

where S_{area} is the area of the selected cell.

5.3.2 Network model

We assume that each sensor user is connected to the nearest eNB. The cell boundaries form a Voronoi tessellation on the plane, as illustrated in Figure 5.1.

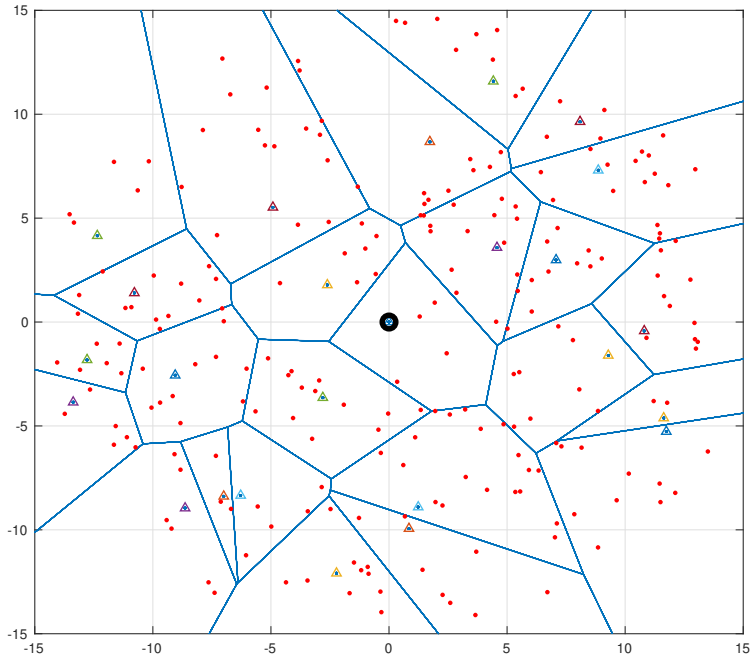


Figure 5.1: A realization of the network modeled by a PPP Φ_b and Φ_u. The typical eNB is denoted by black dot when the sensors by red small dots.

Considering a typical eNB at the origin O. The received power by a sensor located at distance x from O is $P_t x^{-\alpha}/a$, where P_t is the transmitted power from eNB, a is the propagation constant, and α is the path loss exponent. We assume that each eNB allocates Physical Resource Blocks (PRBs) to its sensor users, and each PRB has a bandwidth denoted by W_{rb}.

The user devices in a cell complete to have access to the available number of PRBs, which is denoted by M. Depending on the transmission rate (i.e., the class of services a sensor belongs to) and the position of the sensor user in the cell (i.e., the perceived radio conditions), each user has a given number n PRBs. In this work, we assume that there is just one class of services and denote a required transmission rate by C^*.

The sensor user located at distance x from O is able to decode the signal only if the Signal to Interference plus Noise Ratio (SINR) $\text{SINR}(x) = \frac{P_t x^{-\alpha}/a}{I+\sigma^2}$ is above a threshold θ_{thr}, where I is the received co-channel interference and is assumed to be negligible in our analysis. σ^2 is the thermal noise power. For performance analysis purposes, $\text{SINR}(x)$ is often mapped to the sensor throughput by a link-level curve. The throughput of a sensor located at distance x from O is

$$C(x) = W_{rb} \log_2 (1 + \text{SINR}(x)) \tag{5.2}$$

Moreover, if the SINR is below the critical threshold, then the sensor is said to be in outage and cannot proceed with its communication. Based on the received SINR, one can compute the corresponding maximum coupling loss. The relationship between SINR and MCL is given as follows

$$\text{SINR}_{target} = P_t + 174 - \text{NF} - 10\log_{10}(\text{BW}) - \text{MCL}, \tag{5.3}$$

where NF is the receiver noise figure, and BW is the allocated bandwidth. For the simulation, the same information is repeated n_{rep} times, where n_{rep} being the number of repetitions. The number of repetitions is computed based on different MCL values presented in Table 5.1 [4].

MCL	Repetition
Below 145 dB	1
146 to 148 dB	2
149 to 151 dB	4
152 to 154 dB	8
155 to 157 dB	16
158 to 160 dB	32
161 to 163 dB	64
Above 164 dB	128

Table 5.1: Coverage classes with repetition factor [4].

According to Table 5.1, there are eight classes of users having repetition factors varying from 1 to 128. We assume that each sensor node which has the same number of repetitions attempts to transmit with the same probability $p_l = 2^l/128$, where $l = (0, 1, .., 7)$ represents the class of user corresponding to the repetition factor.

Definition 1: (**Number of resource block**). The eNB may allocate a maximum number N_{max} of resource blocks to each sensor node at each time slot. According to the Shannon formula, a sensor located at a distance x from the center O demanding service of data rate C^*, has the number of PRBs which is defined as follows:

$$\begin{aligned} N(x) &= \min\left(\left\lceil \frac{C^*}{W_{rb} log_2(1 + \text{SINR}(x))} \right\rceil, N_{max}\right) \\ &= \min\left(\left\lceil \frac{C^*}{C(x)} \right\rceil, N_{max}\right) \end{aligned} \tag{5.4}$$

where $\lceil y \rceil$ is the ceiling function of y, W_{rb} is the bandwidth of a resource block, the quantity N_{max} is the maximum number of resource blocks that can be allocated to one sensor user and $C(x)$ is the throughput of a sensor located at a distance x from O.

From (5.4), we realize that the sensor in which a bad channel quality (with a low value of $\mathrm{SINR}(x)$) needs a higher number of PRBs to achieve the transmission rate C^*. Let d_n be the distance from O that verifies, for all $x \in (d_{n-1}, d_n]$, $N(x) = n$, with

$$n = \frac{C^*}{C(d_n)} \tag{5.5}$$

is an integer and

$$d_n = \begin{cases} 0 & \text{if } n = 0 \\ \left[\frac{a(I+\sigma^2)}{P} \left(2^{\frac{C^*}{nW_{rb}}} - 1 \right) \right]^{-\frac{1}{\alpha}} & \text{otherwise.} \end{cases} \tag{5.6}$$

From (5.5), the area of typical cell O can be divided into rings with radius d_n ($1 \leq n \leq N_{max}$, $0 \leq d_{n-1} \leq d_n$). When the sensor requests n PRBs to achieve the transmission rate C^*, the sensor is said to be located in the area between two rings of radius d_n and d_{n-1}. If $x > d_{N_{max}}$, we consider a sensor is out of service and $n = 0$.

In addition, from (5.3) and Table 5.1, we realize that the number of repetitions of each user depends on its SINR value. Let r_l be the distance from O; the typical cell can be divided into rings with radius r_l ($0 \leq l \leq 7, 0 \leq r_{l-1} \leq r_l$). The area between the rings of radius r_l and r_{l+1} defines the region of the cell where the sensors have $n_{rep}(l)$ repetitions ($n_{rep}(l) = 2^l$, $l = 0, 1, .., 7$). For each user having a distance $x \in (r_l, r_{l+1}]$, it has the number of repetitions $n_{rep}(l)$ with:

$$r_l = \begin{cases} 0 & \text{if } l = 0 \\ \left[\frac{a(I+\sigma^2)}{P} \mathrm{SINR}_{\mathrm{target}}(l) \right]^{-\frac{1}{\alpha}} & \text{otherwise.} \end{cases} \tag{5.7}$$

For example, we consider the eight regions in u coverage area. As shown in Figure 5.2, the sensors located in the disk having a radius of r_1 that have one repetition, the sensors that have two repetitions are the sensors in the area that represents the region between the disk $\mathrm{B}(0, r_1)$ and the disk $\mathrm{B}(r_1, r_2)$ and so on.

5.4 Presentation of the NB-IoT dimensioning

Dimensioning process is evaluating the required radio resources that allow serving all user devices given a target QoS. The QoS can be measured by the outage probability metric or even by an average target throughput. In this work, we analyze the impact of the various parameters involved in the dimensioning of the NB-IoT network. This includes environment parameters (fading, path-loss exponent), network and system parameters (cell radius, eNB density, and the active sensor user density), and QoS parameters (capacity and the target value of the outage probability). We illustrate the outage probability as a function of these above key parameters, particularly the number of PRBs threshold M, the eNB and sensor user densities. To characterize this outage probability, we need to evaluate the total number of requested PRBs. In the following, we will present some analytical results regarding the expression of the outage probability under the presented system model.

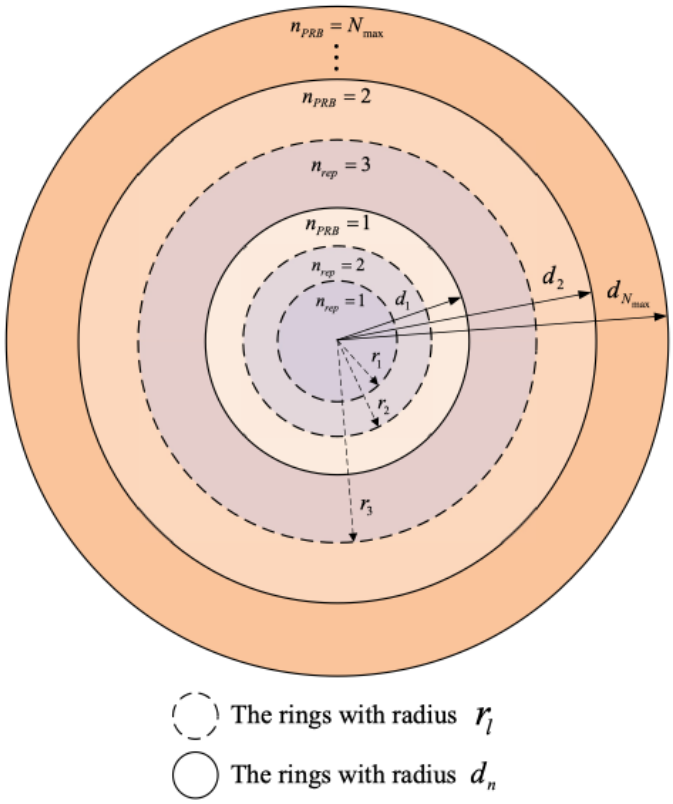


Figure 5.2: Illustration of the cell areas.

5.4.1 The total number of requested PRBs

We propose a definition of "virtual number of users", which is equal to the number of users multiplied by its number of repetitions. To qualify the "virtual number of users", which have n PRBs and n_{rep} repetitions, we consider eight independent Poisson random variables denoted by $X_n^{n_{rep}(l)}$, with parameter $\mu_n^{n_{rep}(l)}$. Note that the sensor users in the same repetition area have the same transmit probability $p_l = 2^l/128$. The mean number of users μ_n that locate between two rings $B(0, d_n)$ and $B(0, d_{n-1})$ is then computed as

$$\mu_n = \sum_{l=0}^{7} \mu_n^{n_{rep}(l)}, \tag{5.8}$$

where

$$\begin{aligned} \mu_n^{n_{rep}(l)} &= n_{rep}(l) p_l \lambda \pi \xi \eta \\ \xi &= d_n^2 \mathbb{1}_{\{r_{l+1} \geq d_n\}} + r_{l+1}^2 \mathbb{1}_{\{r_{l+1} < d_n\}} - (r_l^2 \mathbb{1}_{\{r_l \geq d_{n-1}\}} + d_{n-1}^2 \mathbb{1}_{\{r_l < d_{n-1}\}}) \\ \eta &= \begin{cases} 0 & \text{if } B < 0 \\ 1 & \text{if } B > 0 \end{cases} \end{aligned} \tag{5.9}$$

The number of users that lie between two rings $B(0, d_n)$ and $B(0, d_{n-1})$ is a Poisson random variable denoted by V_n:

$$V_n = \sum_{l=0}^{7} X_n^{n_{rep}(l)} \tag{5.10}$$

Finally, the total number of required PRBs in the typical cell is defined as the sum of demanded PRBs by sensor users in each ring of radius d_n and can be represented as:

$$\Gamma = \sum_{n=1}^{N_{max}} nV_n \tag{5.11}$$

The random variable Γ which is the sum of weighted Poisson random variables, and it is named compound Poisson sum.

5.4.2 The Outage Probability

The outage probability is defined as the probability in which the number of the total request PRBs in the cell is greater than a threshold value fixed by the operator. In other words, it presents the probability of failing to achieve an output number of required PRBs M to satisfy a predefined quality of services $P_{out} = P(\Gamma \geq M)$, where M is the output number of PRBs that required to guarantee a predefined quality of services.

$$\mathbb{P}(\Gamma \geq M) = \mathbb{E}_{\Phi_b}[\mathbb{P}(\Gamma \geq M|\Phi_b)] \tag{5.12}$$

Proposition 1 *Let Λ be a random variable such that $\Lambda = \sum_{n=1}^{N} nV_n$, with V_n are Poisson random variables of intensity w_n. The probability that Λ exceeds a threshold M is*

$$\mathbb{P}(\Lambda \geq M) = 1 - \frac{1}{\pi} e^{-\sum_{n=1}^{N} w_n} \times \int_0^{\pi} e^{p_n(\theta)} \frac{\sin\left(\frac{M\theta}{2}\right)}{\sin\left(\frac{\theta}{2}\right)} \cos\left(\frac{M-1}{2} - q_n(\theta)\right) d\theta, \tag{5.13}$$

where

$$p_n(\theta) = \sum_{n=1}^{N} w_n \cos(n\theta) \quad \textit{and} \quad q_n(\theta) = \sum_{n=1}^{N} w_n \sin(n\theta)$$

Proof. To prove proposition 2, we firstly calculate the moment generating function (i.e., Z-Transform) $f(z)$ of discrete random variable Λ.

$$\begin{aligned} f(z) = \mathbb{E}(z^{\Lambda}) &= \sum_{k=0}^{+\infty} z^k \mathbb{P}(\Lambda = k) \\ &= \prod_{n=1}^{N} \sum_{k=0}^{+\infty} z^{nk} \mathbb{P}(V_n = k) \end{aligned} \tag{5.14}$$

Since V_n is a Poisson random variable with parameter w_n, (5.14) is simplified to

$$f(z) = e^{-\sum_{n=1}^{N} w_n} e^{\sum_{n=1}^{N} z^n w_n} \tag{5.15}$$

It is clear that f is, in particular, inside the unit circle ω. The Cauchy's integral formula gives the coefficients of the expansion of f in the neighborhood of $z = 0$:

$$\mathbb{P}(\Lambda = k) = \frac{1}{2\pi i} \int_{\omega} \frac{f(z)}{z^{k+1}} dz \tag{5.16}$$

After replacing f by its expression in (5.15) and parameterizing z by $e^{i\theta}$, (5.16) leads to

$$\mathbb{P}(\Lambda = k) = \frac{1}{2\pi} e^{-\sum_{n=1}^{N} w_n} \int_0^{2\pi} \frac{e^{\sum_{n=1}^{N} w_n e^{in\theta}}}{e^{ik\theta}} d\theta \tag{5.17}$$

Since the outage probability is defined by the Complementary Cumulative Distribution Function (CCDF) of Λ, we have

$$\begin{aligned} \mathbb{P}(\Lambda \geq k) &= 1 - \sum_{k=0}^{M-1} \mathbb{P}(\Lambda = k) \\ &= 1 - \frac{1}{2\pi} e^{-\sum_{n=1}^{N} w_n} \int_0^{2\pi} e^{\sum_{n=1}^{N} w_n e^{in\theta}} \sum_{k=0}^{M-1} e^{-ik\theta} d\theta \end{aligned} \tag{5.18}$$

After some simplifications from (5.18), we can get the expression of (5.13).

In this work, we introduce a mathematical model called the exponential Bell polynomials [53], [54] to calculate the outage probability. This model is widely used in order to evaluate some integrals and alternating sums. In the following subsection, we recall the definition and some properties of Bell Polynomials.

5.4.3 The exponential Bell Polynomials

The exponential complete Bell polynomials B_p are defined by

$$B_p(x_1, x_2, ..., x_p) = \sum_{k_1 + 2k_2 + ... = p} \frac{p!}{k_1! k_2! ...} (\frac{x_1}{1!})^{k_1} (\frac{x_2}{2!})^{k_2} ... \tag{5.19}$$

and verify the following formula given by the generating function

$$e^{\sum_{j=1}^{+\infty} x_j \frac{t^j}{j!}} = \sum_{p=0}^{+\infty} \frac{t^p}{p!} B_p(x_1, x_2, ...x_p) \tag{5.20}$$

Also, if we consider the following matrix $A_p = (a_{i,j})_{1 \leq i,j \leq p}$ defined by

$$\begin{cases} a_{i,j} = \binom{p-1}{j-i} x_{j-i+1} & \text{if } i \leq j, \\ a_{i,i-1} = -1 & \text{if } i \leq 2, \\ a_{i,j} = 0 & \text{if } i \geq j+2, \end{cases} \tag{5.21}$$

such that

$$A_p = \begin{bmatrix} x_1 & \binom{p-1}{1} x_2 & \binom{p-1}{2} x_3 & \binom{p-1}{3} x_4 & ... & x_p \\ -1 & x_1 & \binom{p-2}{1} x_2 & \binom{p-2}{2} x_3 & ... & x_{p-1} \\ 0 & -1 & x_1 & \binom{p-3}{1} x_2 & ... & x_{p-2} \\ 0 & 0 & -1 & x_1 & ... & x_{p-3} \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & ... & -1 & x_1 \end{bmatrix}$$

then the complete exponential Bell polynomial $B_p(x_1, ...x_p)$ can be defined as the determinant of this matrix.

$$B_p(x_1, ..., x_p) = det(A_p) \tag{5.22}$$

For instance, the first few Bell Polynomials are given by

$$\begin{aligned}
&\hat{B}_0 = 1 \\
&\hat{B}_1(x_1) = x_1 \\
&\hat{B}_2(x_1, x_2) = x_1^2 + x_2 \\
&\hat{B}_3(x_1, x_2, x_3) = x_1^3 + 3x_1x_2 + x_3 \\
&\hat{B}_4(x_1, x_2, x_3, x_4) = x_1^4 + 6x_1^2x_2 + 4x_1x_3 + 4x_2^2 + x_4 \\
&\quad \vdots
\end{aligned} \tag{5.23}$$

Also, the complete Bell polynomials satisfy the binomial type relation:

$$B_p(x_1 + y_1, ..., x_p + y_p) = \sum_{i=0}^{p} \binom{p}{i} B_{p-i}(x_1, ..., x_{p-i}) B_i(y_1, ..., y_i) \tag{5.24}$$

Based on the exponential complete Bell polynomials, we can give the expression of the outage probability as the following proposition.

Proposition 2 *Let Λ be a random variable such that $\Lambda = \sum_{n=1}^{N} nV_n$, with V_n are Poisson random variables of intensity w_n. Let x_j be defined as*

$$x_j = \begin{cases} w_j j! & \text{if } 1 \leq j \leq N \\ 0 & \text{otherwise.} \end{cases} \tag{5.25}$$

The probability that Λ exceeds a threshold M can be expressed as a function of the exponential complete Bell polynomials by

$$\mathbb{P}(\Lambda \geq M) = 1 - H \sum_{k=0}^{M-1} \frac{B_k(x_1, ..., x_k)}{k!} \tag{5.26}$$

where $H = e^{-\sum_{n=1}^{N} w_n}$.

Proof. The moment generating function (i.e., Z-Transform) $f(z)$ of the discrete random variable Λ is calculated as

$$\begin{aligned}
f(z) = \mathbb{E}(z^{\Lambda}) &= \sum_{k=0}^{+\infty} z^k \mathbb{P}(\Lambda = k) \\
&= \prod_{n=1}^{N} \sum_{k=0}^{+\infty} z^{nk} \mathbb{P}(V_n = k)
\end{aligned} \tag{5.27}$$

Since V_n is a Poisson random variable with parameter w_n, (5.27) is simplified to

$$f(z) = e^{-\sum_{n=1}^{N} w_n} e^{\sum_{n=1}^{N} z^n w_n} \tag{5.28}$$

By using the definition of x_j in equation (5.25) and the generating function in equation (5.20), $f(z)$ in equation (5.28) becomes

$$f(z) = He^{\sum_{j=0}^{\infty} z^j \frac{x_j}{j!}} \tag{5.29}$$

where $H = e^{-\sum_{n=1}^{N} w_n}$.

The second exponential term in (5.29) can be evaluated by using the generating function of the complete Bell polynomials given in equation (11); it follows that

$$f(z) = H \sum_{p=0}^{+\infty} \frac{z^p}{p!} B_p(x_1, ..., x_p) \tag{5.30}$$

Using the definition of Z-Transform and the Taylor expansion of $f(z)$ in 0, we get

$$\mathbb{P}(\Lambda = p) = \frac{H}{p!} B_p(x_1, ..., x_p) \tag{5.31}$$

Finally, from the definition of the CCDF, we can obtain

$$\mathbb{P}(\Lambda \geq M) = 1 - \sum_{k=0}^{M-1} \mathbb{P}(\Lambda = k) \tag{5.32}$$

that leads to the equation (5.26).

We apply Proposition 1 with a Poisson random variable V_n of a parameter $w_n = \mu_n$. The outage probability conditionally on Φ_b in (5.12) can be calculated by averaging over the realization of Φ_b:

$$\mathbb{P}(\Gamma \geq M) = \mathbb{E}_{\Phi_b} \left[1 - H \sum_{k=0}^{M-1} \frac{1}{k!} B_k \left(x_1(\Phi_b) .... x_k(\Phi_b) \right) \right] \tag{5.33}$$

where $x_j = \mu_j j!$ and $H = e^{-\sum_{n=1}^{N_{max}} \mu_n}$.

As we mentioned in section 5.3, μ_j is the mean number of sensor users that requested j PRBs with $1 \leq j \leq N_{max}$. In case $N_{max} < j \leq M$ we assume that sensor user j is out of service and $\mu_j = 0$. The outage probability given by equation (5.33) becomes:

$$\mathbb{P}(\Gamma \geq M) = 1 - \left(\hat{B}_0^{(N)} + \hat{B}_1^{(N)} + ... + \frac{1}{(M-1)!} \hat{B}_{M-1}^{(N)} \right) \tag{5.34}$$

where

$$\begin{aligned}
\hat{B}_0^{(N)} &= \frac{1}{N}\sum_{i=1}^{N} H^{(i)} \\
\hat{B}_1^{(N)} &= \frac{1}{N}\sum_{i=1}^{N} H^{(i)} x_1^{(i)} \\
\hat{B}_2^{(N)} &= \frac{1}{N}\sum_{i=1}^{N} H^{(i)} ({x_1^{(i)}}^2 + x_2^{(i)}) \\
\hat{B}_3^{(N)} &= \frac{1}{N}\sum_{i=1}^{N} H^{(i)} ({x_1^{(i)}}^3 + 3x_1^{(i)} x_2^{(i)} + x_3^{(i)}) \\
\hat{B}_4^{(N)} &= \frac{1}{N}\sum_{i=1}^{N} H^{(i)} ({x_1^{(i)}}^4 + 6x_1^{(i)} x_2^{(i)} + 4x_1^{(i)} x_3^{(i)} + 3{x_2^{(i)}}^2 + x_4^{(i)}) \\
&\vdots
\end{aligned} \tag{5.35}$$

Note that for $j > N_{max}$, the value $x_j = 0$.

Once we have the expression of the outage probability, we set a target value P_{out} and then, the required number of available PRBs M is derived as a function of sensor user intensity λ_u through the implicit equation $\mathbb{P}(M, \lambda_u) = P_{out}$. The output M of the implicit function constitutes the result of the dimensioning process.

5.5 Numerical analysis

Numerical analysis is performed on NB-IoT networks, which are generated according to a Poisson Point Process. We explore three scenarios as follows

1. for a given user density λ_u and the transmission rate C, the outage probability is derived in a function of the requested PRBs fixed by the eNB M;

2. for a given eNB density λ_b and the transmission rate C, the outage probability is derived in a function of the requested PRBs fixed by the eNB M;

3. for a given outage probability P_{out}, the required number of available PRBs in the network is derived in the function of the sensor user intensity λ_u;

The proposed model estimates the outage probability in the multi-cell downlink NB-IoT. For numerical purposes, we consider an area of radius $R = 15$ km and each eNB transmits the same power of $P = 40\ dBm$. The sum of the downlink thermal noise power and the receiver noise figure is set to $\sigma^2 = -129,2\ dBm$. For our network model, the propagation parameter is $a = 130\ dB$, and the path loss exponent is considered to be $\alpha = 3.5$. In this work, we consider only one class of service with $C = 80\ kbps$.

Figure 5.3 illustrates the described model in MATLAB for the eNB's density $\lambda_b = 0.9\ eNBs/km^2$ and two values of the active sensor user's density $\lambda_u = 15\ users/km^2$, $\lambda_u = 20\ users/km^2$. We observe that the analytical model of the outage probability fits the result by Monte-Carlo simulation. Moreover, increasing the intensity of active sensors generates an increase in the

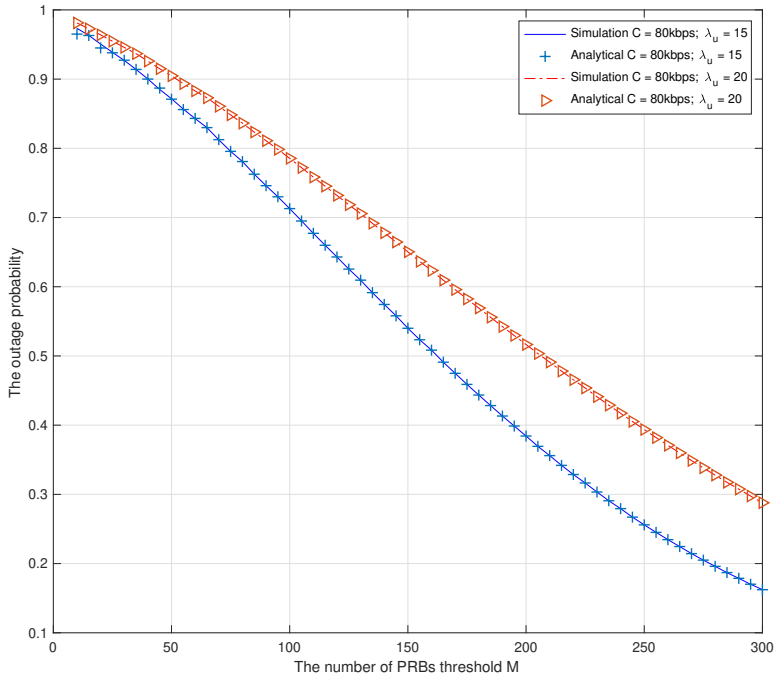


Figure 5.3: The outage probability in analytical and simulation.

outage probability. In other words, the system experiences a high outage when the number of the required PRBs by users increases, resulted from the increasing intensity.

Actually, NB-IoT devices can appear in many different forms: from the indoor devices, such as for gas monitoring and water meters, alerts sensors in building to outdoor devices such as wearable devices for people and animal tracking, health monitoring, the sensors for tracking of attributes of land, pollution, noise, rain in agriculture. Therefore, the signal may be affected by different path loss coefficients. The indoor sensors always need more PRBs to guarantee a required transmission rate because of high path-loss and poor performance in terms of SINR.

To see how eNB's intensity impacts the performance, we plot the outage probability, considering various values of eNB intensities while remaining the active sensor's intensity $\lambda_u = 15\ users/km^2$ in Figure 5.4. We observe that the outage probability decreases when increasing the eNB's intensity. In other words, if the mean number of sensor users that an eNB has to serve is the same, increasing the number of eNBs leads to a decrease in PRB needed in each eNB.

During the resource dimensioning procedure, the eNB starts by defining a target outage probability that can be tolerated for a given service. For a given transmission rate, the number of PRBs is set to ensure that the outage probability never exceeds its target value. Figure 5.5 presents the number of required PRBs to ensure the target value of the outage probability ($P_{out} = 1\%$ and $P_{out} = 5\%$) with a fixed transmission rate of 80 $kbps$. We can observe that for the target value of outage probability, the number of resource blocks required in the cell increases when the sensor user intensity increases.

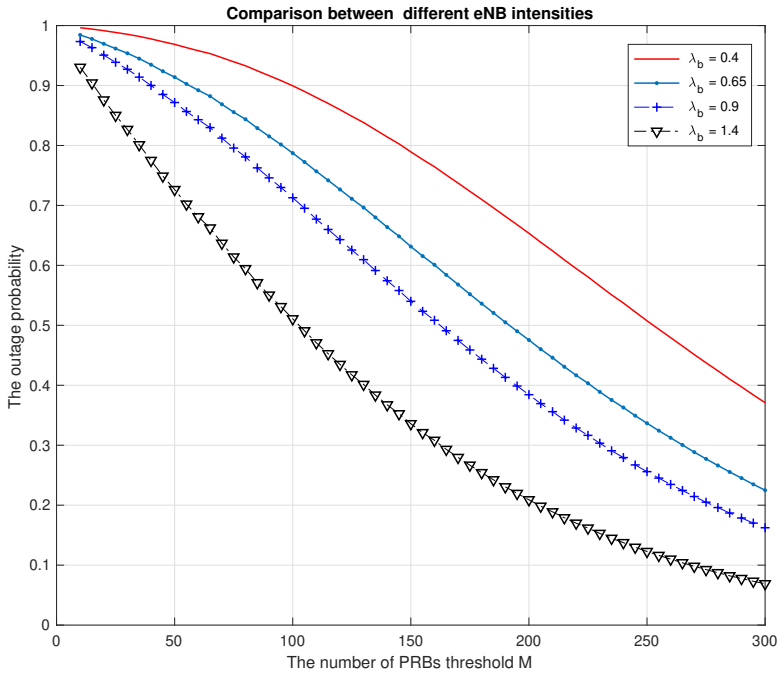


Figure 5.4: Comparison between different eNB intensities.

5.6 Conclusion

In this Chapter, we have presented a resource dimensioning model for NB-IoT network. We considered a multi-cell NB-IoT downlink network in which sensor nodes and eNBs are distributed according to a Poisson point process. We have derived an analytical model taking into account the repetition of data transmission to determine the number of required PRBs as a function of sensor user's intensity and eNB intensity. The simulation results show that the proposed model is accurate. Moreover, we have established an implicit relationship between the required resources and the sensor users intensity given a target outage probability. This relationship translates the dimensioning problem that an operator can perform to look for the number of necessary radio resources to satisfy a predefined QoS. In future works, we will investigate the impact of the channel conditions such as shadowing and interference to the network system.

5.7 Resulting research contribution

The research contribution resulting from the work done in this chapter is below:

- T. A. Nguyen and P. Martins, "Multi-cell Downlink Dimensioning in NB-IoT Networks," 2020 16th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Thessaloniki, Greece, 2020, pp. 117-122.

  doi: 10.1109/WiMob50308.2020.9253401.

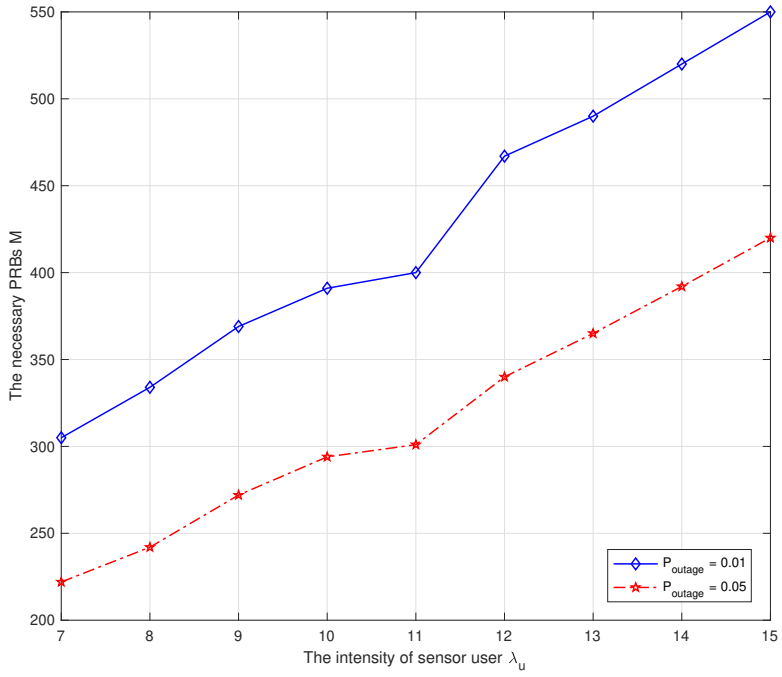


Figure 5.5: Required PRB M as a function of sensor's intensity.

Chapter 6

Traffic Pattern Recognition and Prediction

6.1 Introduction

Starting from Long Term Evolution (LTE), the concept of Self-Organizing Networks (SONs) has been introduced by 3GPP Release 8 [9] and intends to decrease network deployment times, reduce congestion, improve throughput, and reduce the installation and management costs. However, several methods provided for SONs, such as the Markov model or genetic algorithms, are both inefficient and costly for mobile operators. Most of these solutions require manual data analysis and adjustment of the system parameters to optimize the network. Besides, through the functionalities of self-configuration, self-optimization, and self-healing, SONs focus on increasing network automation and thus minimize human intervention in cellular network management. This objective can be done with a Deep Learning approach.

In recent years, Machine Learning (ML) is one of the most potential and promising approaches for analyzing complex problems. ML techniques are able to learn from the data sets and improve performance over training time. The result is the execution of a given task, such as classification, prediction, or clustering. Machine learning has been widely applied in the case of image processing or speech recognition. Large amounts of datasets exist where researchers can apply various methods. However, there is a major limitation for the advancement of Machine learning-based research in the wireless and mobile networks area due to the lack of extensive datasets. Despite those difficulties, the importance of deep learning in the mobile networking domain has been recognizing and has some ability to solve specific problems in the LTE or the Fifth-Generation (5G) Mobile communications systems.

Toward offering a wide set of cellular services in energy and cost-efficient way, the network needs to be constructed based on predictive analysis of mobile network traffic and proactive network resource management. Network operators must use the traffic planning tools so they can know in advance about the state of future traffic demands. Recognition of data traffic patterns and prediction of the type of packet arrivals will be a key job that SON performs. A key feature aimed at supporting technology development and the ability to respond to a wide variety of traffic types is demonstrated by the ability of networks to perform automatic traffic analysis through classification tools. The classification of network traffic into appropriate classes has many relevant uses, such as Quality of Service (QoS) control, network resource management, malware detection, and intrusion detection. The key challenge

of such classification algorithms consists in the identification of traffic type through a number of representative features. These features are used to train algorithms that recognize the traffic pattern. The development of deep learning techniques has recently paved the way for automatic feature extraction and modelization with high performance [55]. Besides, traffic prediction has many applications in many networking areas, such as energy-saving, network resource management, etc. Proactive knowing the traffic pattern and demands are being used to save a significant amount of power. If the network traffic can be predicted accurately, the processors in the sensor devices can be turned off during low traffic times to save power. It is necessary in order to power up the NB-IoT devices and optimize energy saving. A precisely predicted future traffic pattern will help us design an efficient and green network. In addition, traffic pattern recognition and prediction are required to use efficiently network resources to ensure good quality of service. As a result, in recent years, there has been an increasing interest in leveraging machine learning tools in analyzing the aggregated traffic for optimizing the operation of the network [56, 57]. Analysis of cellular traffic for finding traffic anomalies provoked by a rapid growth in the number of users when a crowded event takes place nearby the monitored area has been investigated in [55].

It is important to remark that almost all the approaches presented in the current state of the art implement traffic volume prediction and classification based on determined traffic sessions. While most of the traffic prediction problems focus on capturing the dynamic of traffic and enhancing the performance, there is a lack of research on the analysis and understanding at the user side. Unlike the previous approaches, this work proposes to use a Deep learning model by using information extracted from the Downlink Control Information (DCI). This information is used as time-series sequence data for the training process and is used to recognize and predict the mobile traffic patterns. By applying the Long Short-Term Memory (LSTM), we can get efficient recognition and prediction performances. Specifically, the recognition task achieves an accuracy up to $98,3\%$, and the prediction task ensures a loss lower than 10^{-1}.

6.2 Related Works

We summarize state-of-the-art researches on cellular traffic prediction and recognition and introduce the research gaps which motivate our work.

Traffic classification/recognition As mentioned in the Introduction, the use of Deep learning techniques has recently applied to mobile networks and led to superior performance. Traffic classification has been studied in the network domain with three main approaches such as UDP-TCP port-based, deep packet-inspection and statistical based [58]. First, *UDP-TCP port-based* analysis relies on the fact that most Internet applications use Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) port numbers. The port numbers can be easily accessed and are usually not affected by encryption, thus allowing a fast flow classification. In [59], the authors propose an automated classifier generation system based on destination IP addresses and port numbers for application-level mobile traffic identification. It should be noted that the port-based approach hardly work when the applications use dynamic port number. In addition, some modern applications flow with unregistered port numbers are the cause of diminishing of the port-based classifier performance. Second, *the deep packet inspection* (DPI) concept consists of analyzing the captured packet contents to accurately and timely classify the traffic generated by different Internet protocols. The most used technique relies on inspecting packet contents and matching them with a deterministic

set of patterns. In [60], the authors present a survey of payload-based traffic classification approaches. Most important DPI analysis are presented, including the evaluation of the classification accuracy, through a common set of traffic traces and the computational requirements. In [61], the authors propose a maximum entropy-based IP-traffic classification scheme to extract the application layer payload pattern. The proposed scheme achieves high accuracy for Support Vector Machine (SVM), Naive Bayes, and partial decision trees algorithms. Remarkably, payload-based methods are limited by a high computational cost and load on the classification device. Furthermore, examining an encrypted packet with this method is impossible. Inspecting the contents of a packet poses privacy challenges, and this act may represent a breach of security and privacy policies. *Statistical classification* is a technique that exploits statistical characteristics of network traffic flow to identify the application. Each type of application has a unique measurement to distinguish different applications from each other. In [62], the authors classify the service usage using encrypted Internet traffic in mobile messaging applications by jointly modeling user behavioral patterns, network traffic characteristics, and temporal dependencies. The packet length-related features and the time delay-related features from traffic-flows are extracted to prepare the training data. The effectiveness of the classifier in statistical-based classification depends on the features extracted from the data, which require extensive knowledge due to their complexity [58]. Besides, since this approach does not deal with packet contents so it can work with encrypted traffic without any difficulty.

We note that most of the reviewed works classify mobile traffic based on manual feature extraction and work with network or application-level data classification. We realize that there is a need for research of the cellular traffic pattern recognization scheme without looking into the packets (due to encryption and latency) and without analyzing the inter-packer arrival for all packets (due to latency and complexity).

Traffic prediction To meet the strict requirements of the next-generation network and its new set of applications, the network needs to understand the dynamics of traffic and predict the future user's demands. The prediction of mobile traffic patterns has been studied through time-series analysis methods due to the characteristics of the datasets. The most popular used methods, such as Auto Regressive Integrated Moving Average (ARIMA) [63] and Support Vector Regression (SVR) [64] have been applied to capture the dynamic of the mobile traffic. However, the limitation of such methods is the poor capturing of the rapid fluctuations in the time series. ARIMA reproduces the time series patterns based on the average of the past value and fails to accurately predict traffic patterns in highly dynamic data such as in cellular networks. The SVR methods are also limited because there is no available way to determine the best parameters of the model. In addition, these methods only work well with the homogeneous time series, where the input and the output are within the same set of values. The accuracy performance is not achieved well when there is a high number of metrics in the dataset, or the dataset structure is complex. In recent years, Artificial Neural Networks (ANNs) and Recurrent Neural Networks (RNNs) have found application in network traffic prediction, in particular, when the dataset consists of multivariate features and presents heterogeneous. ANN is promised as an excellent tool to find the complex patterns in input data and has been used for the prediction of network traffic [65]. In [66], RNN has been used to design a spatio-temporal prediction model for cellular networks with multi-task learning. In [67], researchers have proposed to used neural network models on cellular traffic to analyze trade activities in urban commercial districts. Moreover, Long Short-Term Memory (LSTM), which is an implementation of RNN, allows to solve the vanishing gradient problem and can handle the long term dependencies of the data. In [63], LSTM structures have been proposed

to study a spatio-temporal dataset of the mobile traffic from different base stations.

Reviewing the state-of-the-art reveals that there is a lack of research of leveraging deep learning models for cellular traffic pattern predictions, selection of adequate features, which are extracted directly from the decoded Download Control Information (DCI) message.

6.3 Deep Learning Architectures

This section presents an overview of several deep learning architectures that have been used to solve the problem of mobile traffic characterization in this Chapter. We will recap the RNN and LSTM theory and explain why we selected LSTM as our primary method.

6.3.1 RNN

A Recurrent Neural Network (RNN) utilizes sequential information in which the output depends not only on the current inputs but also on the previous inputs. They are called recurrent because the data is similarly processed for every element in the data sequence. Because of their internal memory, RNNs can remember important information about their inputs and thus are preferred for time series data.

RNN shares the parameters for every element of a sequence and generates outputs that depend on the current and previous inputs. It uses hidden states to hold information on previous input. Figure 6.1 illustrates the RNN architecture and its unfolding form. Here

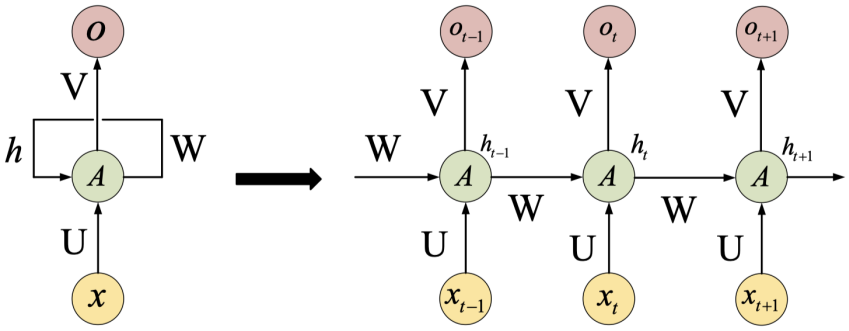


Figure 6.1: An unfold of recurrent neural network.

$x = (x_0, x_1, x_2, ..., x_t)$, $o = (o_0, o_1, o_2, ..., o_t)$ and $h = (h_0, h_1, h_2, ..., h_t)$ represent the input, output and hidden state series respectively. A hidden state h_t receives information from the previous hidden state h_{t-1} and the current input x_t, acting as the memory of the network that keeps the information about what was previously computed. The output o_t is calculated based on the memory at time t. The parameters involved in a RNN are described as follows.

$$\begin{aligned} h_t &= \tanh\left(U.x_t + W.h_{t-1}\right) \\ o_t &= \text{softmax}\left(V.h_t\right) \end{aligned} \tag{6.1}$$

where U is the weight matrix between the input x_t and the hidden state h_t, W is the weight between the previous hidden state h_{t-1} and the current one h_t, V is the weight between the hidden state h_t and the output o_t.

Training a RNN network is similar to training a traditional Neural Network using the backpropagation algorithm. In RNN, the parameters are shared by all timesteps in the network; the gradient at each output depends not only on the error calculation of the current step but also on the previous time steps. This is a reason called Backpropagation Through Time (BPTT).

Nevertheless, the range of the historical information preserved by the hidden states is limited, which is known as a gradient vanishing problem.

Except for the time series forecasting problem, RNN were rarely applied for time series classification due to three main factors:

- RNN's architecture is designed mainly to predict output for each timestamps in the time series [68];
- RNNs typically suffer from the problem of vanishing gradients, which prevents the training of long data sequences [69];
- RNNs are considered hard to train, which led the researchers to avoid using them for computational reasons [70].

The limitations of RNN networks

1. **Long Term Dependencies** When the gap between the relevant information and the present task is small, RNN networks can learn to use the past information to perform the present task, as presented in Figure 6.1. But when the gap grows, RNNs become unable to learn to connect the information. This problem is studied in [71], [72].

2. **Vanishing gradient problem**

   As more layers containing activation functions are added, the gradient of the loss function approaches zero. The gradient descent algorithm finds the global minimum of the cost function of the network. The network with many hidden layers can cause the gradient to be too small for model training.

   Considering the effect of vanishing gradients on the derivatives of a cost E_t at time $t = 3$ with weights W

$$\frac{\partial E_3}{\partial W} = \sum_{k=0}^{3} \frac{\partial E_3}{\partial \hat{y}_3} \frac{\partial \hat{y}_3}{\partial \hat{s}_3} \frac{\partial s_3}{\partial s_k} \frac{\partial s_k}{\partial W} \tag{6.2}$$

   where $\frac{\partial s_3}{\partial s_k}$ is a chain rule, i.e., $\frac{\partial s_3}{\partial s_1} = \frac{\partial s_3}{\partial s_2} \frac{\partial s_2}{\partial s_1}$. The result of (6.2) is a Jacobian matrix whose elements are all the pointwise derivatives. The gradient value in (6.2) can be rewritten as:

$$\frac{\partial E_3}{\partial W} = \sum_{k=0}^{3} \frac{\partial E_3}{\partial \hat{y}_3} \frac{\partial \hat{y}_3}{\partial \hat{s}_3} \left(\prod_{j=k+1}^{3} \frac{\partial s_j}{\partial s_{j-1}} \right) \frac{\partial s_k}{\partial W} \tag{6.3}$$

The gradient value in (6.3) can be considered as an absolute value; of the above Jacobian matrix has an upper bound of 1 [73]. The output activation function of RNN is tanh or sigmoid, which have derivatives of 0 at both ends. When the derivative equal to 0, the corresponding neurons are saturated. At that time, the gradients in previous layers will also be saturated. The gradient values will explode very fast with small values in the matrix and multiple matrix multiplications; even its will vanish after a few steps. Thus, the state at "far away" steps will no longer work with the current node, making it impossible for the RNN to learn long-range dependencies. This problem not only occurs with RNN but also in deep Feed-forward Neural Network.

There are a few ways to avoid the vanishing gradient problem. One of them is using ReLu instead of tanh or sigmoid activation functions. An even more popular solution is using Long Short-Term Memory (LSTM), which was designed to avoid vanishing gradients and efficiently learn long-term dependencies.

6.3.2 LSTM

The Long Short Term Memory (LSTM) is a particular kind of Recurrent Neural Network (RNN). LSTM is designed to avoid the long-term dependency problem, which is the cause of the vanishing gradient problem, and cannot be solved in RNN. This network structure was originally introduced by Hochreiter et al. [74], and has been refined as a powerful technique to handle the problem of time series predictions [75]. The network structure of LSTM is similar to that of standard RNN. The main difference is that LSTM has long short-term memory blocks that consist of memory cell units. These memory cell units make it possible for LSTM to remember state values for an arbitrary long time. A LSTM block also has three different gate units that learn to keep, utilize, or remove a state when appropriate. Each of three gates can be considered as regulators of the flow of values that goes through the connections of the LSTM.

A typical LSTM network is comprised of different memory blocks call cells, as shown in Figure 6.2. The key of LSTMs is the cell state, the horizontal line running through the top of the diagram. The LSTM has the ability to remove or add information to the cell state, regulated by structures call gates. There are three gates in each LSTM cell to protect and control the cell state, i.e., a forget gate, an input gate, and an output gate. When the information enters the LSTM network, the LSTM cells will decide to block or pass the information. Only important information will be left. The unimportant information will be discarded by the forgotten gate, thus solving the long-term dependence problem of standard RNN.

Forget gate

The forget gate is used to decide which information from the previous cell could be passed to this cell. This gate takes in two inputs: h_{t-1} and x_t, which are the hidden state from the previous cell (or the output of the previous cell) and the input of the current cell at that particular time step, respectively. The given inputs are multiplied by the weight matrices W_f, and a bias b_f is added. Following this, the sigmoid function is applied to this value. The sigmoid function outputs a vector, with values ranging from 0 to 1, corresponding to each number in the cell state. Basically, the sigmoid function is responsible for deciding which values to keep and which to discard. The cell will forget this information if f_t is zero and

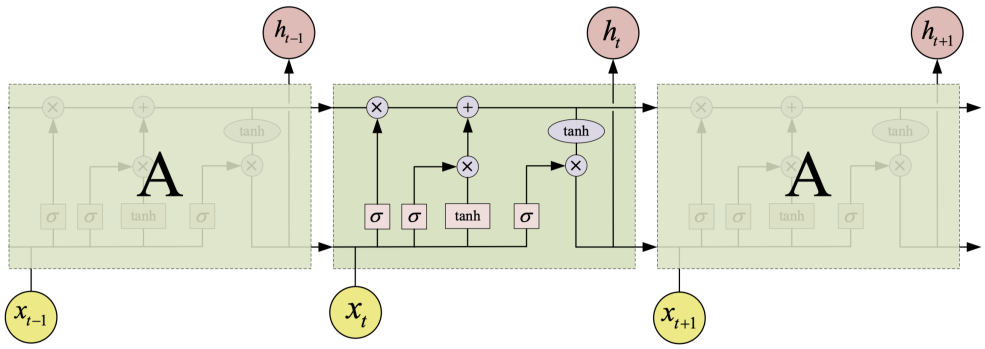


Figure 6.2: The architecture of LSTM network.

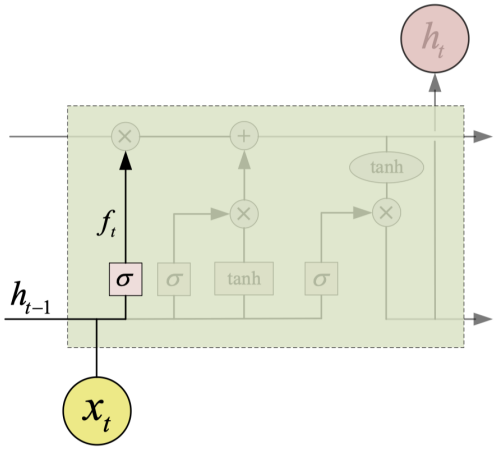


Figure 6.3: Forget gate in a LSTM cell.

remember otherwise.

$$f_t = \sigma\left(W_f\left[h_{t-1}, x_t\right] + b_f\right) \tag{6.4}$$

This vector output from the sigmoid function is multiplied to the cell state.

Input gate

The input gate decides which values will be updated in this cell. This addition of information is basically three-step process, as shown in Figure 6.4.

- The sigmoid function layer will decide values which will be updated, i.e., i_t calculated in (6.5). This is basically very similar to the forget gate and acts as a filter for all information from h_{t-1} and x_t;

- Creating a vector of new candidate values, i.e., $\tilde{C}_t$ as perceived from h_{t-1} and x_t to the cell state. This is done using the **tanh** function, which output values from -1 to $+1$.

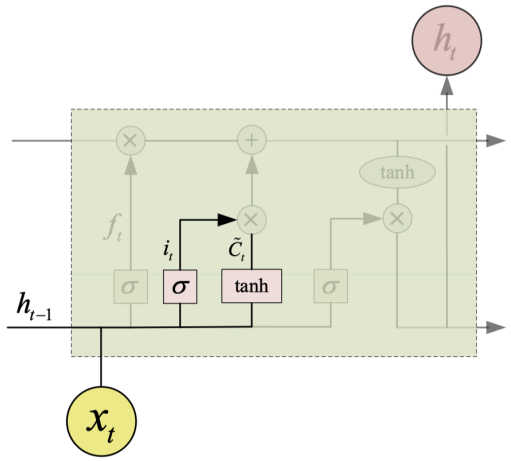


Figure 6.4: Input gate in a LSTM cell.

- Multiplying the value of the regulatory filter (the sigmoid gate) to the created vector (the tanh function) and then adding this useful information to the cell state via addition operation.

$$\begin{aligned} i_t &= \sigma\left(W_i\left[h_{t-1}, x_t\right] + b_i\right) \\ \tilde{C}_t &= \tanh\left(W_C\left[h_{t-1}, x_t\right] + b_C\right) \end{aligned} \tag{6.5}$$

By this way, the information is added to the cell state that is important and is not redundant.

To update the old cell state C_{t-1} into the new cell state C_t as described in (6.6), we multiply the old state by f_t, forgetting the things we decided to forget earlier, then add $i_t\tilde{C}_t$. This is the new candidate values, scaled by how much we decided to update each state value.

$$C_t = f_t C_{t-1} + i_t \tilde{C}_t \tag{6.6}$$

Output gate

Finally, the cell has to decide the output value by the output gate. The functionality of an output gate is three-step process as shown in Figure 6.6.

- Creating a vector after applying tanh function to the cell state, thereby scaling the values to the range -1 to +1.
- The inputs h_{t-1} and x_t pass through the sigmoid layer to decide which parts of cell state being output as o_t presented in (6.7).
- Multiplying o_t to the vector created in the first step and sending it out as an output and also to the hidden state of the next cell.

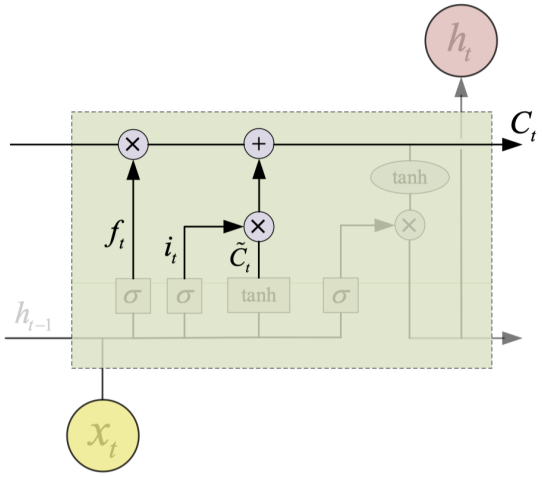


Figure 6.5: Update state in a LSTM cell.

$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right) \tag{6.7}$$
$$h_t = o_t \tanh C_t$$

Multiple LSTM cells are concatenated to form one layer of the LSTM network, as shown in Figure 6.2. Each cell computes the operations on one time index and transfers the output to the the next LSTM cell. LSTM neural network has been successfully applied to applications with sequential data such as classify, process and predict time series given time stamps. Multiple layers of basic LSTM cells form a stacker LSTM network. The LSTM cell of each layer extracts a fix number of features which are passed to the next layer. The number of layers represents the depth of the network, which is to increment the accuracy of the prediction, which is done by the last fully connected layer.

For the recognition problem, we use a many-to-many architecture, which means that the network observes the traffic sample for a fixed number of timeslots until T and try to recognize the traffic patterns at time slot T. For the prediction problem, we use a many-to-one architecture and try to predict the traffic patterns in the next time slot $T + 1$.

6.4 Dataset Collection

In recent years, Software-Defined Radio (SDR) is an important emerging technology in wireless communications, in which some of the radio functions typically implemented in hardware are converted into software on a personal computer or embedded system. There are some LTE network emulators that offer a software platform, developed based on the standardized LTE protocols, namely Amarisoft, Open Air Interface (OAI), and srsLTE. Like Open Air Interface (OAI), Amarisoft LTE supports full stack LTE protocol in software in terms of complete implementation and emulation capabilities. In [4], the authors conduct a detailed security analysis on different mobile networks, including Global System for Mobiles (GSM), General Packet Radio Service (GPRS), and Long Term Evolution (LTE). An end-to-end mobile communication security testbed using open source components on a virtualized platform is set up. To implement the LTE network, LTE Amarisoft 100 is used as the EPC in the testbed.

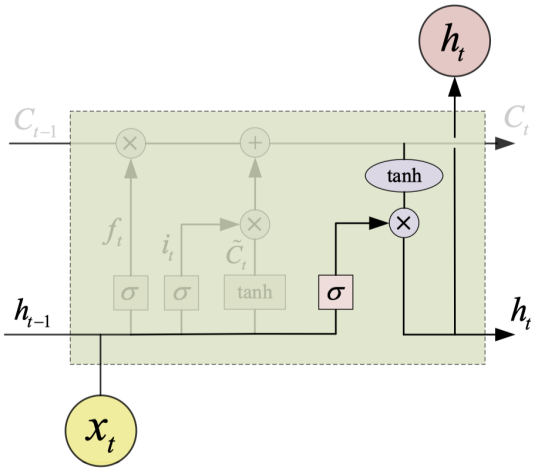


Figure 6.6: Output gate in a LSTM cell.

Authors in [5] propose a mobile relay architecture for the public transport system. They evaluate the architecture with Amarisoft software suite.

In this thesis, we use the Amarisoft LTE 100 as the EPC in our testbed. This technology contains all LTE components such as EPC, eNB, NB-IoT, evolved Multimedia Broadcast Multicast Service (eMBMS) and IP Multimedia Subsystem (IMS), as shown in Figure 6.7.

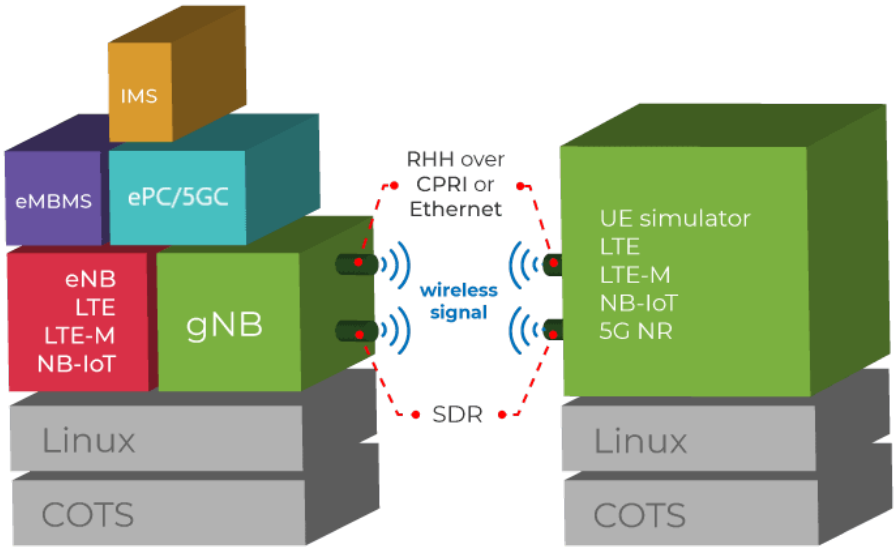


Figure 6.7: The Amarisoft technology [3].

LTEENB, a module of Amarisoft, allows building a real LTE base station using a standard PC and a low-cost software radio front-end. All the physical and protocol layer processing is done in real-time inside the PC, so no dedicated LTE hardware is necessary. It can be used to set up an entire LTE open network [76]. Figure 6.8 presents the architecture of our test-bed. The two gray blocks represent the different computers that are involved in the experimental

process. The *Test Controller* coordinates and controls all other blocks, such as the Amarisoft instances, the USB attenuator. It is connected to the *Backhaul Computer* via Ethernet and to UEs via USB links. Two Amarisoft programs running on the backhaul computer:

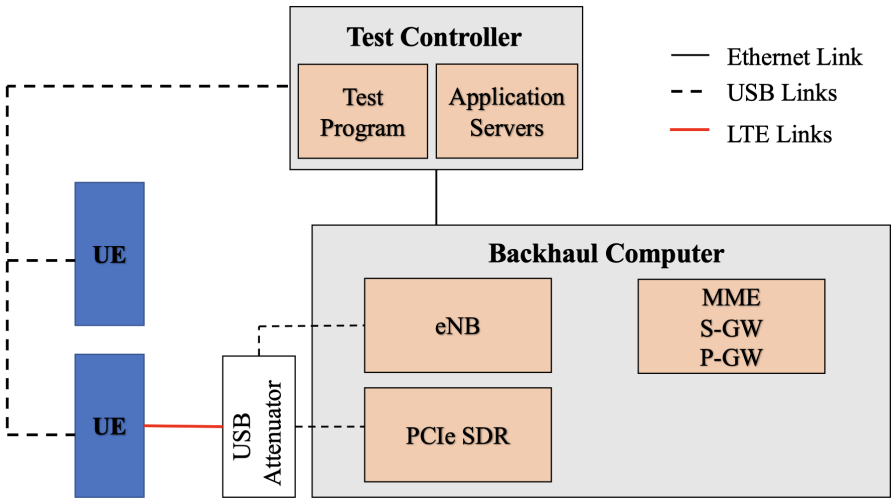


Figure 6.8: The Amarisoft test-bed architecture.

- The *ltemme* program runs to manage the Mobile Management Entity (MME), Serving Gateway (SGW), and Packet Gateway (PGW) required to handle the UEs. It is connected to a wired network, allowing to route traffic from the UEs to application servers.

- The *lteenb* program manages the eNB. It is a LTE base station (eNodeB) implemented entirely in software and running on a PC [27]. The PC generates a baseband signal which is sent to a radio front end doing the digital to analog conversion. The reverse is done for the reception. LTE-ENB interfaces with a LTE Core Network through the standard S1 interface. In particular, the Amarisoft Core Network software (LTEMME) can easily be connected to it to build a highly configurable LTE test network.

The mobile traffic data used in this work comes from the Amarisoft LTE Web GUI, in which allows to analyze LTE software logs and get real-time information from the system [3]. An illustration of the Web GUI interface for logging and analysis is shown in Figure 6.9.

The simulation in Web GUI allows the creation of different types of IP traffic simulations. Each simulation will involve several User Equipments (UEs) generating traffic according to their probability. We are interested in studying the traffic exchanged between the eNodeB and all the users in this work. We have generated our cellular traffic dataset and made part of it available online [77]. The study focuses on four following features, which are obtained from the log file to describe mobile traffic communication.

- $\mathrm{BR_{DL}}$: Downlink (DL) bitrate in bits per seconds;

- $\mathrm{RB_{DL}}$: Average number of DL resource block allocated;

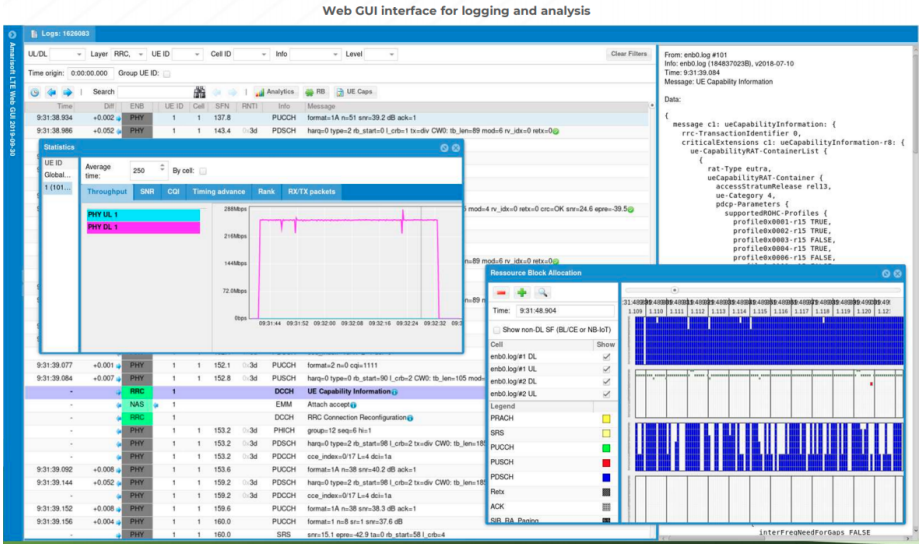


Figure 6.9: WEB GUI interface for logging and analysis.

- $\mathrm{BR}_{\mathrm{UL}}$: Uplink (UL) bitrate in bits per seconds;
- $\mathrm{RB}_{\mathrm{UL}}$: Average number of UL resource block allocated.

Let T be the total measurement period, for every second $t \in T$, we defined $\mathbf{x}(t) = [x_1(t), x_2(t), x_3(t), x_4(t)]$ as the vector that contains all above information, i.e., $\{x_1, x_2, x_3, x_4\} = \{\mathrm{BR}_{\mathrm{DL}}, \mathrm{RB}_{\mathrm{DL}}, \mathrm{BR}_{\mathrm{UL}}, \mathrm{RB}_{\mathrm{UL}}\}$. Therefore, the sequence $\mathbf{X} = [\mathbf{x}(t)]_{t=1}^{T}$ is a multi-variate time-series including the metrics described above, which are gathered from the DCI message through Amarisoft interface.

A labeled dataset is obtained by running specific configurations at user equipment (UE) under our simulation scenario, detecting its C-RNTI within the PDCCH channel and finally associating the corresponding DCI trace with a label, which links it to the traffic application that is executed at the UE.

In this study, we collect 1414 traffic data points with 1 second sampling. There are four labels according to the traffic patterns, including UDP, HTTP, VoIP, and Signalling. Therefore, the original dataset contains an input matrix $\mathbf{X}$ and a vector $\mathbf{y}$ of labels. The dataset is then conveniently pre-processed to be fit with different deep learning model. For the sake of clarity, the procedures for recognition and prediction tasks will be depicted in each below subsection.

6.5 The proposed framework

6.5.1 Research problem description

The state-of-the-art studies show that packet-level and flow-level characteristics extracted from traffic flows are discriminating features to give their application patterns. Therefore, the objective of mobile traffic recognition can be formulated as a traffic pattern classification problem.

Suppose $\{c_1, ..., c_M\}$ are the classes of traffic patterns, a recognition model can be trained to classify the class label of any given traffic flow, i.e., $\Phi(\mathbf{x}) = c_i \in \{c_1, ..., c_M\}$, where $\mathbf{x}$ is the input feature vector of the traffic flow with dimension D, i.e., $\mathbf{x} = (x_1, x_2, ..., x_D)$. The training process takes as input a labeled dataset $DS = (\mathbf{x}(i), y(i)), i = 1, ..., T$, where $\mathbf{x}(i)$ and $y(i)$ are the feature vector and traffic pattern label of the i-th timestamp respectively. For example, suppose we are given a training dataset with three traffic patterns including UDP, HTTP, and VoIP, i.e., $y(\mathbf{x}) \in \{\text{UDP, HTTP, VoIP}\}$. A supervised classification model can be built to accurately discriminate the class label in the set of three known traffic patterns. Besides, a proposed prediction model can provide the traffic type at the next timestamp. We aim to train a multi-class model that classifies a test flow to one of the target classes, i.e., $\Phi_c(\mathbf{x}) \rightarrow \{c_i, \bar{c}_i\}$. Alternatively, we propose a prediction model, where one-step future traffic can be predicted to belongs to the particular class, i.e., $\Phi_p(\mathbf{x}) \rightarrow c_i$. The approach solving these problems is illustrated in Figure 6.10.

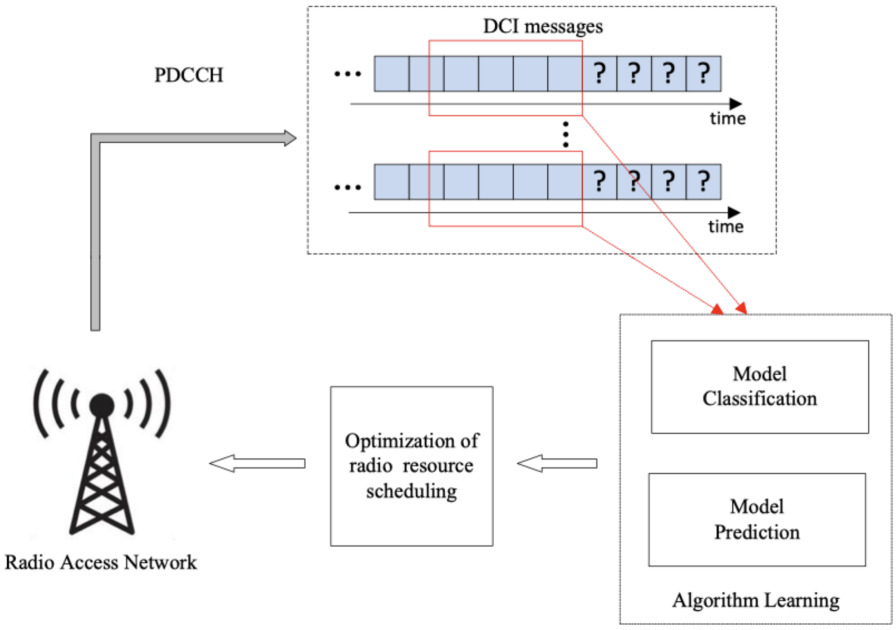


Figure 6.10: The proposed module for traffic pattern recognition and prediction.

The research problem tackled in this work could be stated as follows: given the data collected from DCI messages, how accurately we can estimate:

- The type of traffic to be served (i.e., traffic pattern recognition);
- The prediction of traffic type in a future time interval? (i.e., traffic pattern prediction).

We address the problems of traffic pattern identification and prediction using RNNs, which are one of the state-of-the-art deep learning techniques to deal with temporally correlated data. In particular, we use LSTM cells to build different deep learning architectures. In the first architecture, we implement the stacked-LSTM, which has the task of learning to identify traffic patterns from the input samples. In the second design, we use LSTMs with Autoencoder to predict the traffic pattern at the next time-instant. A discussion on the performance of the

algorithms is provided in the evaluation sections. The achieved results show the proposed deep learning frameworks' capabilities to accurately identify and predict the traffic pattern from the input data samples.

6.5.2 Cellular Traffic Pattern Recognition

The problem of discriminating the network traffic pattern is considered a multi-task classification problem. In this work, we design a classification system based on stacked Long Short-Term Memory (LSTM) neural networks capable of tracking long-term dependencies from multi-variate time-series and solving the vanishing gradient problem that affects standard RNNs. A stacked LSTM network can extract the temporal dependencies of the mobile traffic patterns and learn to discriminate the difference between them.

The approach that we use in this work is supervised. The problem is addressed as a multi-task classification problem, where the designed algorithm is in charge of classifying the cellular traffic patterns into classes: *UDP*, *VoIP*, *HTTP*, and *Signalling*. The whole framework to solve the recognition problem is depicted in Figure. It takes as input data downloaded from DCI, and it consists essentially of 2 parts: *Data Preprocessing* and *LSTM Recognizer*. The implementation detail of each part is discussed in the below subsections.

Data preprocessing

Before being input into the recognition algorithm, the cellular traffic dataset needs to be preprocessed. The input matrix $\mathbf{X}$ describes the captured information of D different metrics for the amount of time equal to T. Thus, the matrix $\mathbf{X}$ has a dimension equal to $T \times D$, where T and D are the number of rows (time instants) and the number of columns (traffic information features) respectively in X. The vector $\mathbf{y}$ of labels contains the traffic type of the time instant, with a dimension equal to $T \times 1$. For example, given the time slot i, it holds that $\mathbf{x}_{i,j} \in \mathbf{X}$ and $y_i \in \mathbf{y}$ are the traffic information transfers during the i-th time slot and the label describing the traffic pattern of the i-th time slot. The multi-variate input cellular traffic time-series $\mathbf{X}$ is performed by the following two preprocessing steps:

- *Data normalization*: The sequence $\mathbf{X}$ is standardized by removing the mean and scaling to unit variance. This operation is performed to normalize the original sequence to reduce the variance of the input dataset.

- *Data windowing*: The data windowing procedure has been depicted in Figure 6.11. The sequence $\mathbf{X}$ is split and grouped using a window length W, which defines how many timestamps are processed by the LSTM network to recognize the traffic type of the input. After splitting, we have $N = T - W + 1$ sequences $\tilde{\mathbf{x}}(n), n \in [1, N]$. A sequence $\tilde{\mathbf{x}}(n)$ has length W, and each of its elements is D-dimensional with $D \in [1, 4]$, the number of considered metrics as described above. Hereafter, to keep simple in the notation, we keep using $\mathbf{X}$ also to represent the pre-processed data. A new matrix $\mathbf{X}$ has a dimension of $N \times W \times D$ is considered as input to the LSTM model.

  The vector $\mathbf{y}$ is used to generate a new set of labels, namely $\mathbf{l_r}$, describing the traffic type associated to the last element in sequence $\tilde{\mathbf{x}}$. It means that by considering W steps before time t, we can recognize the traffic type at the considering instant t. The vector $\mathbf{l_r}$ has a dimension of $N \times 1$.

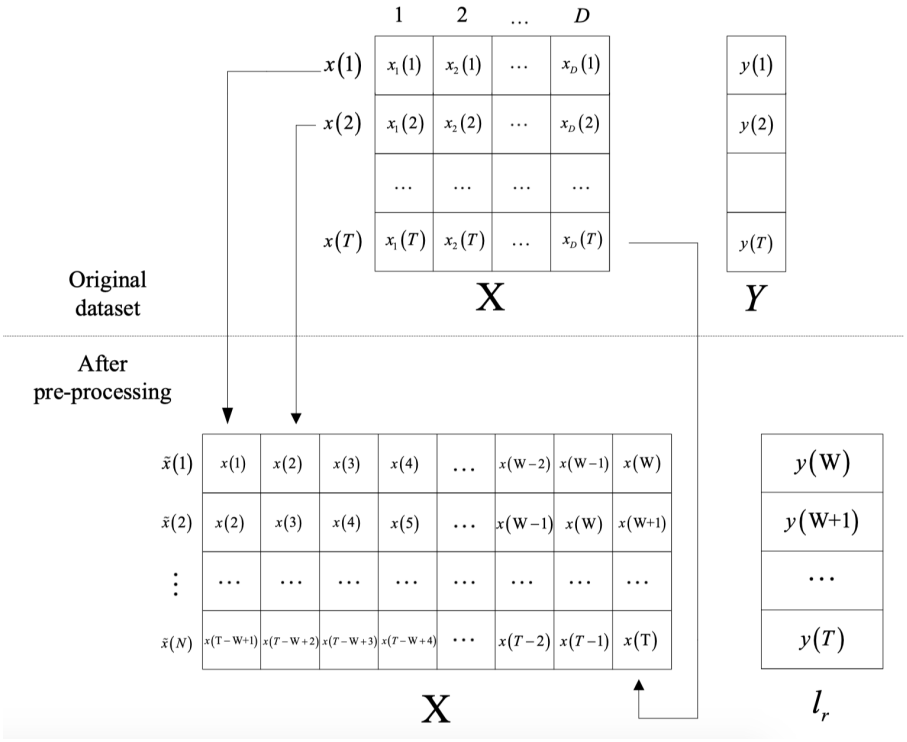


Figure 6.11: Pre-processing of the training dataset.

- *One-hot-encoding*: Each sample can belong to one of M classes. This process turns each possible type of traffic into a vector. The vector size is the number of classes in the dataset, i.e., M. Its components all are 0, except for a single 1 at the component representing a particular class.

Finally, 70% of the dataset is used as a training set, while the remaining 30% is used as a testing set.

LSTM Recognizer

We use the stacked-LSTM architecture that includes multiple LSTM layers to make traffic pattern recognition. The number of concatenated cells in the first layer indicates the number of observations of the data, which corresponds to the window length W. The LSTM recognizer estimates a function $\Phi_c : \mathbf{X} \rightarrow Y$ that for each traffic sequence $\tilde{\mathbf{x}}$ at time n, $\tilde{\mathbf{x}}(n) = [\mathbf{x}(n), \mathbf{x}(n+1), ..., \mathbf{x}(n+W-1)]$, the recognizer tries to estimate the traffic type at time $n+W-1$, $\tilde{\mathbf{y}}(n+W-1)$.

$$\Phi_c = (\mathbf{x}(n)) = \tilde{\mathbf{y}}(n+W-1) \tag{6.8}$$

To describe the traffic type of the investigated timestamp, the recognizer uses the softmax layer, based on the softmax activation function, working with a number of classes (i.e., the considered traffic types). Note that our methodology can be extendable to any number of

classes. The softmax layer of the recognizer is configured by penalizing the categorical cross-entropy (CCE) loss function between true label y_i and the learned label $\tilde{y}(i)$. CCE is a loss function for a multi-class recognition task [78], which is defined in (6.9)

$$\text{CCE} = -\sum_{k=1}^{M} y_{ik} \log(\tilde{y}_{ik}) \tag{6.9}$$

where M denotes the number of classes, y_{ik} is the class labels associated with the input $\tilde{\mathbf{x}}(n)$, and $\tilde{y}_{ik}$ is the recognition output obtained for input $\tilde{\mathbf{x}}(n)$.

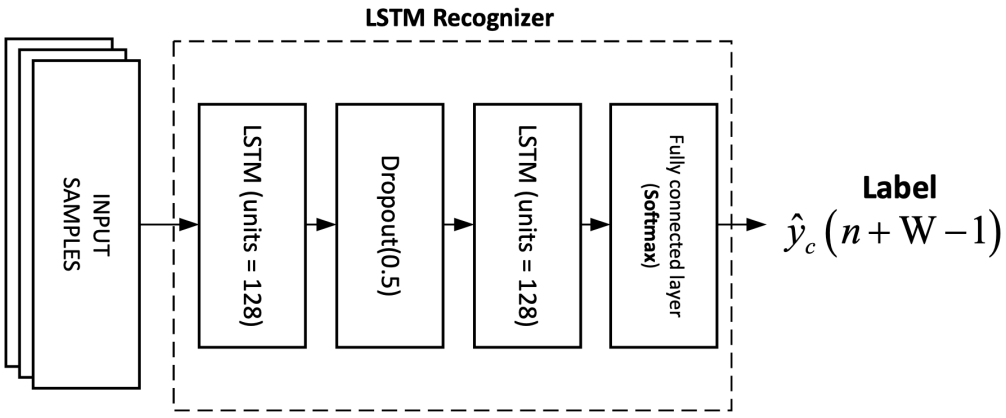


Figure 6.12: The recognition model.

Evaluation Metrics

Regarding performance metrics, we adopt the overall accuracy A_c. In addition, for each traffic pattern, we derive the per-class metrics including precision P, recall R, and F-score F.

- *The recognition accuracy* A_c quantifies the percentage of correctly recognized with respect to the total number of recognitions made.

$$A_c = \frac{\text{number of correct recognition}}{\text{Total number of recognitions made}} \tag{6.10}$$

- *Precision* P is the ratio between true positive T_p (the number of samples belonging to that class that is correctly classified) and the sum between true positives and false positives F_p, where F_p represents those samples incorrectly classified as belonging to that class.

$$P = \frac{T_p}{T_p + F_p} \tag{6.11}$$

- *Recall* R describes the ratio between the true positives T_p and the sum between true positives and false negatives F_n, which are the samples from that class that are incorrectly classified as belonging to another class.

$$R = \frac{T_p}{T_p + F_n} \tag{6.12}$$

- *F-score* is defined as the harmonic mean of precision P and recall R.

$$F = \left(\frac{\frac{1}{P} + \frac{1}{R}}{2} \right)^{-1} = 2\frac{RP}{R+P} \tag{6.13}$$

In addition, we use a confusion matrix to present the results of the proposed model in an unambiguous way. This is a useful metric that presents both class distribution in each data label and what types of errors the recognizer is making.

Performance Evaluation

The recognition algorithm has been implemented in Python using Google Colab, which provides free hardware acceleration with Tensor Processing Unit (TPU). We used *keras* library on top of Tensorflow backend for the implementation of RNNs. For the training part, we use the Adam algorithm as an optimizer and categorical cross-entropy as loss functions.

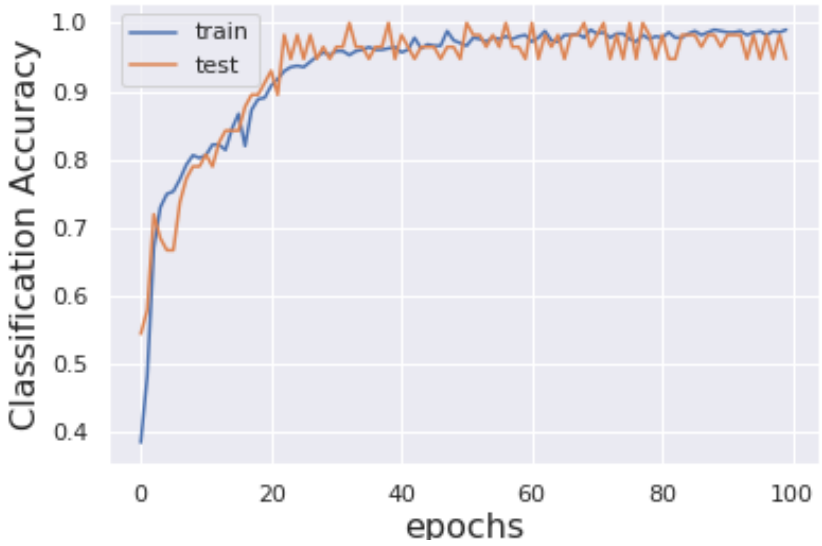


Figure 6.13: The accuracy of recognition task.

Figure 6.13 illustrates the recognition accuracy of the proposed model as a function of the number of epochs. The performance improves after each epoch. The accuracy obtained is greater than 90% after 20 epochs. It can be observed that it is possible to reach an accuracy level up to $98,25\%$ for the recognition task.

Moreover, we compare the proposed LSTM scheme with several existing schemes, including K-Nearest Neighbors (KNN), Random Forest Classifier (RF), Multilayer Perceptron (MLP), One-vs-Rest Support Vector Machine (OvR SVM), and One-vs-One (OvO). The overall accuracy of the different learning schemes are presented in Figure. The proposed learning scheme LSTM outperforms the existing schemes by successfully identifying 90% of the testing classes. The increases in accuracy indicate that the traffic pattern extraction scheme is able to help the learning systems capture the correlation between the applications from the control channel information.

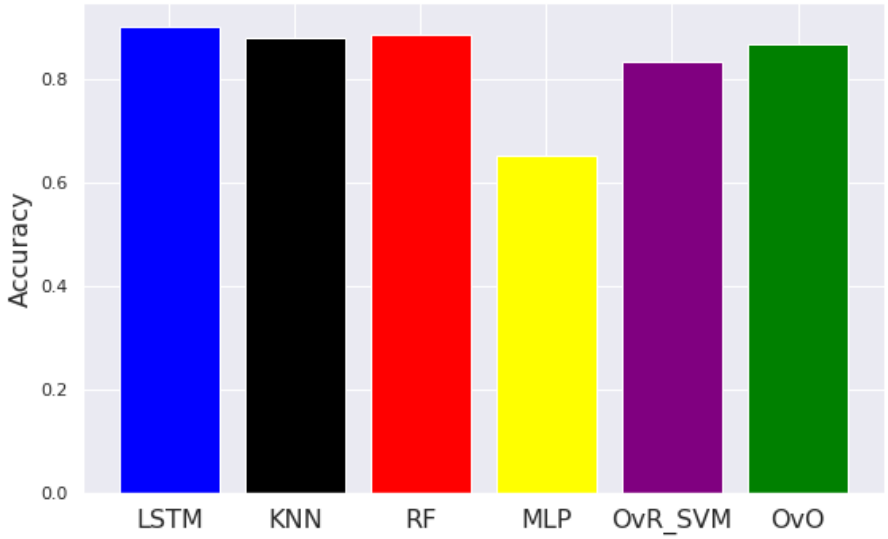


Figure 6.14: Overall accuracy results.

The confusion matrix with normalization by class support size (number of elements in each class) is provided in Figure 6.15 to analyze which types are mismatched in the recognition process. There are still wrong recognizations between Signalling, HTTP, and UDP traffic types. In fact, 11% and 12.5% of HTTP and UDP are wrongly recognized as Signalling while 2% and 3.6% of Signalling are wrongly classified as HTTP and UDP, respectively. Obtained results further confirm that similar "patterns" of traffic types affects the recognition performance, especially with HTTP and UDP traffics.

The per-class classification performance is given in Figure 6.16 in terms of (a) precision, (b) recall, (c) f1-score of the four traffic classes. We analyze the performance for each learning scheme in turn. First, the Multilayer Perceptron is incapable of recognizing some traffic patterns; the traffic can not be classifier to the UDP and VoIP classes, thus decreasing their precision rates. For example, an almost number of UDP are put to the VoIP class, resulting in a precision of 0; a lot of Signalling traffics are classifier to VoIP class, resulting in a recall of 0.58. The precision for HTTP class is not affected that much. These sum up to the f1-measure results of 0 for UDP, 0.52 for VoIP, 0.68 for Signalling, and 0.935 for HTTP.

All other classic schemes result stable performances for Signalling and HTTP classes. For UDP, the recall of OvR and OvO schemes are 0.72 and 0.90, but the precision rates are just 0.64 and 0.6. By investigating the confusion matrix details, we find that a lot of UDP traffics are put in Signalling in OvR scheme. These errors result in the low f1-score measure for UDP at around 0.67.

Overall, the proposed LSTM scheme derives better per-class performance than the classical schemes. It has the ability to recognize different types of traffic evenly, especially outperforms the existing schemes by successfully identifying 96% of the VoIP traffic. The outstanding performance of LSTM will also be shown in the traffic type prediction problem in the following section.

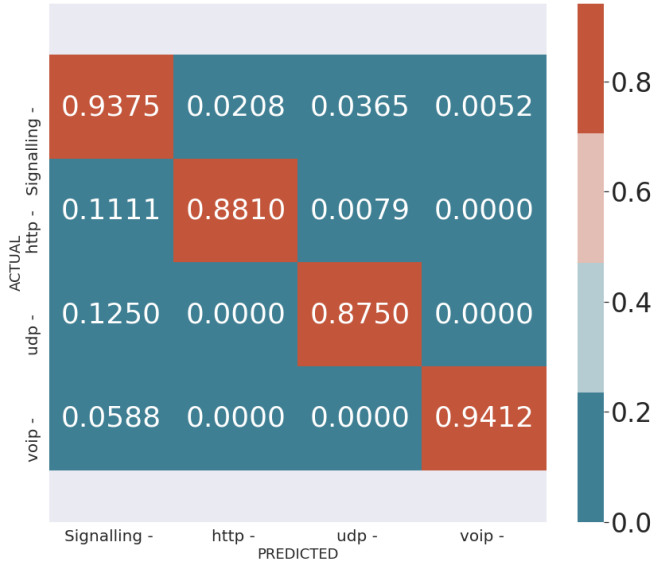


Figure 6.15: Confusion matrix in the recognition task.

6.5.3 Cellular Traffic Pattern Prediction

Prediction of future traffic type demands is a crucial requirement for improving resource efficiency. Instead of trying to identify the type of input samples as presented in the previous section, the objective now becomes to predict the traffic type in the next time-instants. Therefore, the idea is that the algorithm is taught to predict the upcoming traffic pattern, where the prediction error is expected to be low. This work will provide a deep learning framework, which can be applied to optimize network resource allocation and specifically scheduling.

The framework to solve this prediction problem consists of 2 parts: Data Preprocessing, the Autoencoder, and LSTM Predictor. The implementation detail of each part is discussed in the below subsection.

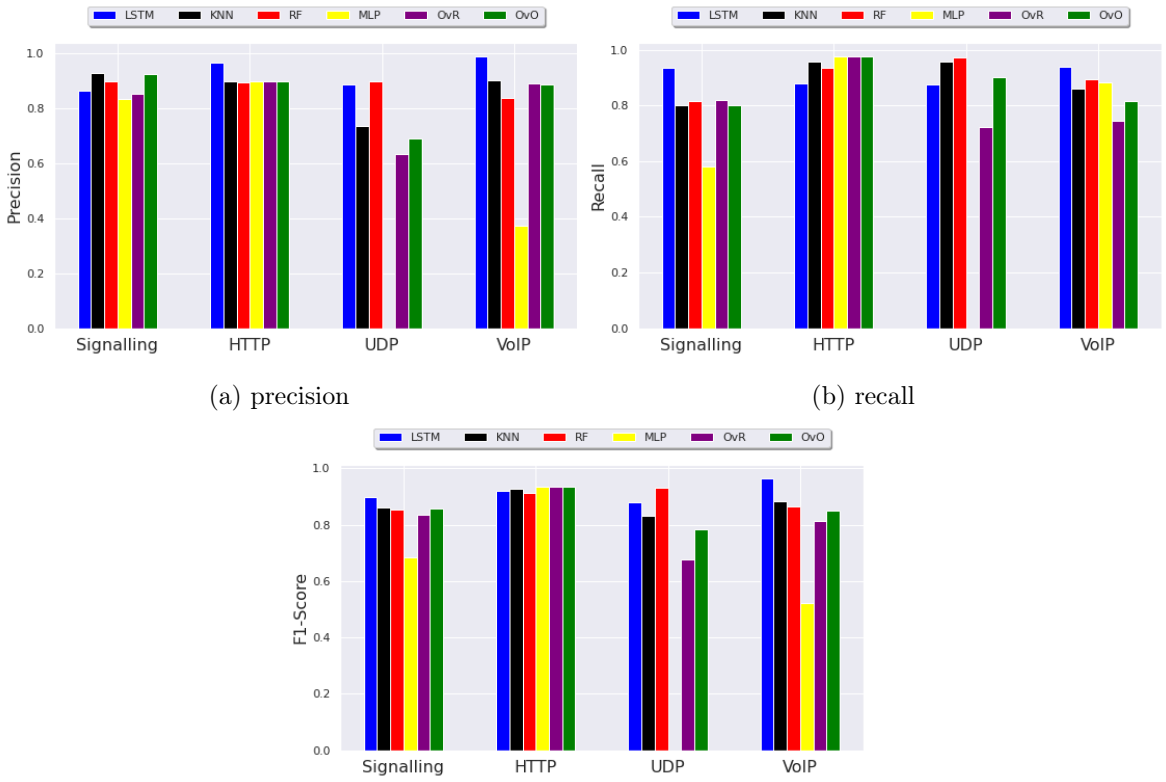
Data preprocessing

The matrix $\mathbf{X}$ and a vector of label $\mathbf{y}$ need to be preprocessed before being input to the LSTM predictor. The data processing procedure consists of two steps:

- *Data normalization* Before feeding into the network, the time-series data $\mathbf{X}$ is normalized by scaling all features in the same range. This process is achieved by min-max normalization and ranges output data between 0 and 1 as follows:

$$\mathrm{x}_{norm} = \frac{\mathrm{x} - \mathrm{x}_{min}}{\mathrm{x}_{max} - \mathrm{x}_{min}} \tag{6.14}$$

- *Data windowing* The normalized sequence $\mathbf{X}$ is then split and grouped using a window length W, which defines the number of timestamps that the LSTM architecture processes



(a) precision

(b) recall

(c) F-score

Figure 6.16: Per-class metric results in terms of (a) precision, (b) recall, and (c) F-score.

to predict the traffic label in the upcoming timestamp. After splitting, we have $N = T - W + 1$ sequences $\tilde{\mathbf{x}}(n), n \in [1, N]$ as follows:

$$\begin{aligned} \tilde{\mathbf{x}}(1) &= [\mathbf{x}(1), \mathbf{x}(2), ..., \mathbf{x}(W)] \\ \tilde{\mathbf{x}}(2) &= [\mathbf{x}(2), \mathbf{x}(3), ..., \mathbf{x}(W+1)] \\ &\cdots \\ \tilde{\mathbf{x}}(N) &= [\mathbf{x}(N), \mathbf{x}(N+1), ..., \mathbf{x}(T)] \end{aligned} \tag{6.15}$$

Each element in a sequence $\tilde{\mathbf{x}}(n)$ has D variables. From now, we note the sequence $\tilde{\mathbf{x}}$ as samples. The matrix $\mathbf{X}$ with dimension $N \times W \times D$ is considered as input to the LSTM model. The vector label $\mathbf{y}$ is used to generate a new set of labels, namely $\mathbf{l_p}$, describing the traffic type in the next timestamp. The dataset of the label is also converted to a set of the one-hot vector by the One-hot-encoding procedure.

Finally, 70% of $\mathbf{X}$ and $\mathbf{l}_p$ is used as a training set, while the remaining 30% is used as a testing set.

The Autoencoder

The autoencoder represents a type of Artificial Neural Network (ANN) used to learn efficient data codings in an unsupervised manner. The objective of an autoencoder is to learn a representation for a set of data by training the network to ignore signal "noise". In our case, we implement a sequence-to-sequence autoencoder, which manages the encoder through LSTM cells [79]. The encoding representation generated by the encoder is provided to the predictor for comparison with the case without an encoder.

LSTM Predictor

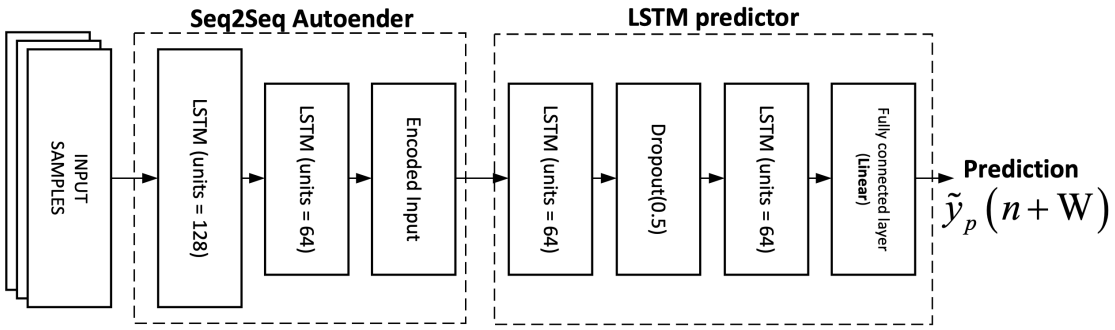


Figure 6.17: The prediction model.

We use a many-to-one architecture for single-step prediction, which means that the network observes the mobile traffic for a duration W until T and then tries to predict the traffic type in the next time slot $T + 1$. Figure 6.17 presents the LSTM architecture to make the traffic type prediction. Our proposed model consists of a stacked architecture that includes multiple LSTM layers. The prediction algorithm Φ_p is taught to predict the traffic type where the prediction error is expected to be low. The proposed algorithm receives a traffic sequence of

length W: $\tilde{\mathbf{x}}(n) = [\mathbf{x}(n), \mathbf{x}(n+1), ..., \mathbf{x}(n+W-1)]$, and tries to predict the traffic type at time $n+W$, $\tilde{y}(n+W)$

$$\Phi_p\left(\tilde{\mathbf{x}}(n)\right) = \tilde{y}_p(n+W) \tag{6.16}$$

The predictor is configured in order to minimize the Mean Squared Error (MSE) loss function, which is determined by

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \tag{6.17}$$

where n is the number of data points, y_i is the true value of encoded traffic type in sample i, and $\tilde{y}_i$ denotes the predicted value of encoded traffic type.

It is expected that the prediction loss function will increase with the time distance between the latest value of the investigated traffic profile and the predicted traffic pattern.

Performance Evaluation

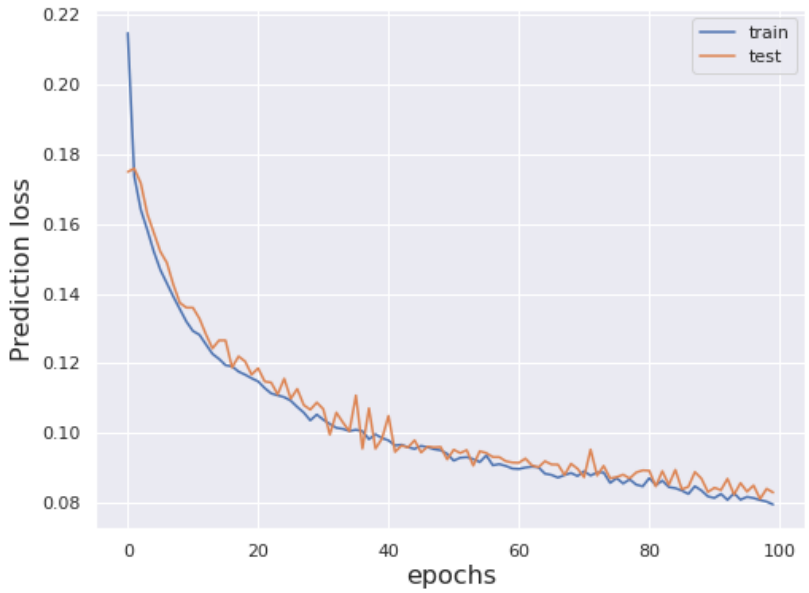
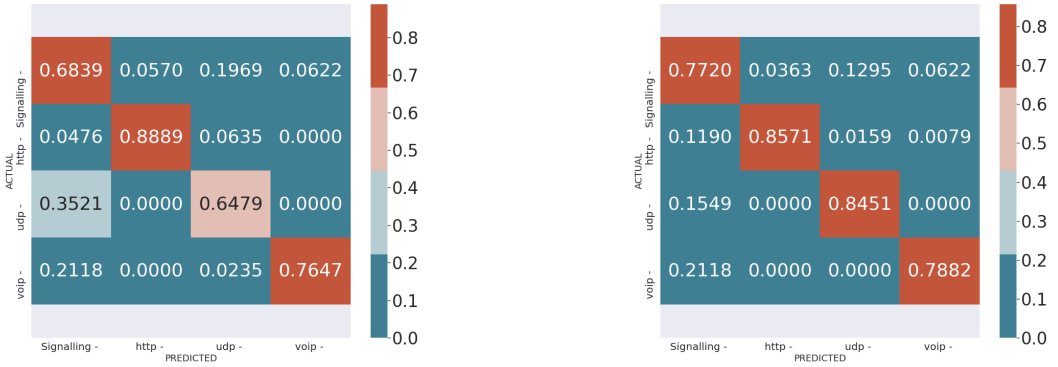


Figure 6.18: The prediction loss performance.

The prediction algorithm has been implemented in Python using Google Colab, which provides free hardware acceleration with Tensor Processing Unit (TPU). We used *keras* library on top of Tensorflow backend for the implementation of RNNs. For the training part, we use the Adam algorithm [80] as an optimizer and the MSE as loss functions.

To show the convergence speed of the proposed model, the training and testing losses after each epoch are given in Figure 6.18. It is possible to observe that the loss decreases quickly during the first 30 epochs and then achieves low-performance loss of 0.08 after 90 epochs.



(a) Without Autoencoder (b) With Autoencoder

Figure 6.19: Confusion matrices in the prediction task

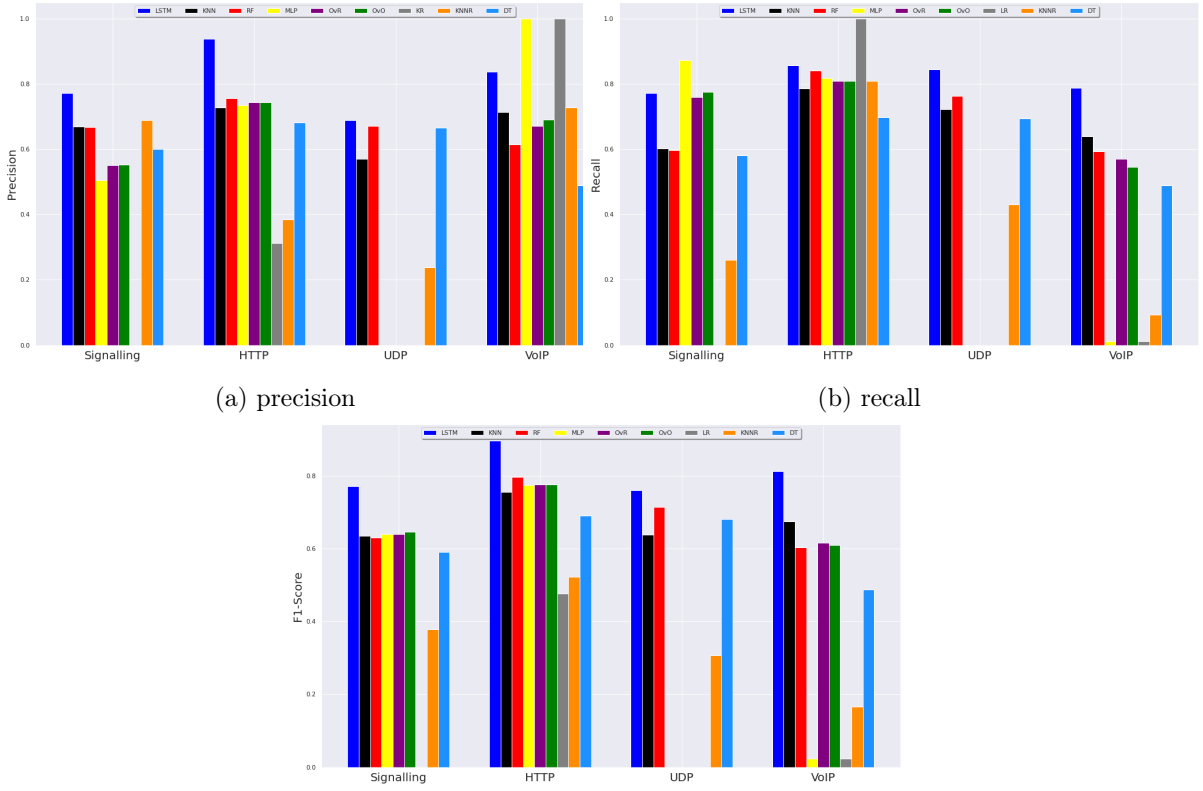
The obtained results show the ability of LSTM to process time-series data for traffic type prediction suitably.

The advantage of applying the autoencoder into the prediction model is presented by considering two confusion matrices in Figure 6.19. Using an autoencoder, the proposed prediction approach can clearly classify the traffic types. In fact, 35.21% of UDP-traffic is wrongly predicted as Signalling. This percentage decreases to 15.49% when using an autoencoder. In addition, in this case, it is possible to observe how the autoencoder approach provides better results with respect to this measure for the simple prediction without an autoencoder approach.

We also present a performance comparison between the proposed LSTM algorithm with several existing algorithms, including the regression algorithms and the classification algorithms. The per-class performance given in Figure 6.20 shows that the regression algorithms such as Linear Regression (LR), KNeighbors Regressor (KNNR), and Decision Tree Regressor (DT) are incapable of predicting the traffic patterns in a future time step.

In addition, if we consider the prediction problem as the classification problem in a future step, we can apply the classical classification algorithms, such as KNeighbors Classification (KNN), Random Forest (RF), Multilayer Perceptron (MLP), OnevsRest (OvR), and OnevsOne (OVO) to predict the traffic patterns. The MLP we derived by using five hidden layers of 100 nodes; each setting is biased towards the positive class. This reflects in the per-class metrics as it obtains perfect 1.0 precision for VoIP, along with extremely low recall lower 0.02 and unsatisfactory f1-score 0.02. It can not predict the UDP traffic type at all. Moreover, it can be observed that OvR and OvO algorithms can not predict the UDP (prediction equal 0), resulting in all UDP traffic is mispredicted with Signalling with the prediction of 0. The achieved performance on VoIP is only around 0.6.

In contrast, the proposed LSTM algorithm completes the task in quite a different way. First, it successfully predicts most of the traffic types, for which the precision rates are between 0.78 and 0.93. Second, the f1-scores for UDP and VoIP outperform all other algorithms and achieve 0.76 and 0.81, respectively. Third, a small amount of traffics from all classes is also falsely predicted as well as shown in Figure 6.19. These false predictions could be caused by



(a) precision

(b) recall

(c) f1-score

Figure 6.20: Per-class metric results in terms of (a) precision, (b) recall, and (c) F-score.

an imbalance in the dataset or in a state transition between Signalling and other traffic types.

6.6 Multi-class multi-output traffic pattern recognition

Section 6.5 has presented the multi-class traffic pattern recognition and prediction, in which the input can be classified into only one class within a pool of classes. In the wireless communication system, the assumption of traditional classification that each sample only belongs to one class is not applicable. The primary reason is that the traffic samples from wireless communication are extremely complicated, and each sample could be labeled as a set of non-binary properties. Besides assigning an appropriate label set for one traffic sample to represent its characteristics, the classifier needs to derive the corresponding number of each label. This type of recognition problem is also called multi-class multi-output classification.

6.6.1 Problem description

In contrast to traditional classification, in this section, we extend to present a multi-class multi-output traffic recognition model, which allows the classification of an input traffic sample into more than one label in the pool of labels and predicts the number class of each label. In multi-class multi-output traffic pattern classification, the task becomes extremely challenging because each traffic sample can be assigned multiple pattern labels. One traffic sample can be represented by a vector of features and a label set rather than by one label exclusively.

The multi-class multi-output classification problem can be defined as follows: Let $\mathcal{X} = \mathbb{R}^D$, Label $= \{w_1, ..., w_L\}$ and n is the possible number class of each label. Therefore, the possible number $N_{\mathcal{Y}}$ of label set $\mathcal{Y}, Y \subseteq \mathcal{Y}$ is the L-permutations of n:

$$N_{\mathcal{Y}} = \frac{n!}{(n-L)!} \tag{6.18}$$

The main problem leads to the classification of "type of traffic" and "number of each traffic type appearing" in a traffic flow. Assume the type of traffic has the possible classes: "UDP", "HTTP", "VoIP", the number of each traffic type in a flow varies from 0 to 4. Then, the number of output is 60.

We denote a dataset composed of N multi-labeled samples by $\mathcal{D} = \{(\mathbf{x}(i), Y(i))\}_{i=1}^{N}$ in (6.19).

$$\mathcal{D} = \{(\mathbf{x}(i), Y(i))\}_{i=1}^{N} = \begin{bmatrix} x^1(1) & x^2(1) & \dots & x^D(1) \\ x^1(2) & x^2(2) & \dots & x^D(2) \\ \vdots & \vdots & \ddots & \vdots \\ x^1(N) & x^2(N) & \dots & x^D(N) \end{bmatrix} \begin{bmatrix} Y(1) \\ Y(2) \\ \vdots \\ Y(N) \end{bmatrix} \tag{6.19}$$

where $\mathbf{x}(i) = [x^1(i), ..., x^D(i)] \in \mathcal{X}$ is the representation of a data sample, $Y(i) = [y_1(i), ..., y_L(i)] \in \mathcal{Y}$ is output label set, which is presented in (6.20)

$$\begin{bmatrix} Y(1) \\ Y(2) \\ \vdots \\ Y(N) \end{bmatrix} = \begin{bmatrix} y_1(1) & y_2(1) & \dots & y_L(1) \\ y_1(2) & y_2(2) & \dots & y_L(2) \\ \vdots & \vdots & \ddots & \vdots \\ y_1(N) & y_2(N) & \dots & y_L(N) \end{bmatrix} \tag{6.20}$$

Multi-class multi-output learning aims to build a classifier $\Phi_m : \mathcal{X} \rightarrow \mathcal{Y}$ using training set $\mathcal{D}$ and optimizes some evaluation metrics.

6.6.2 Dataset

The mobile traffic data used in this section is extracted from the Amarisoft LTE Web GUI, which is described in Section 6.4. The study focuses on the following features, which are obtained from the log file.

- RNTI: the total number of the assigned user equipment;
- $\mathrm{BR_{DL}}$: the total downlink bitrate in bit per second;
- $\mathrm{BR_{UL}}$: the total uplink bitrate in bit per second;
- $\mathrm{RB_{DL}}$: the total number of resource blocks allocated in the downlink;
- $\mathrm{RB_{UL}}$: the total number of resource blocks allocated in the uplink;

We indicate with D the number of metrics considered in $\mathbf{x}(t)$. Therefore, the sequence $\mathbf{x}(t)$ is a multi-variate time-series, which includes the metrics extracted directly from the decoded DCI messages and aggregated over all the assigned user equipment. The label set $\mathcal{Y}$ is also obtained from the log file from the Amarisoft Web GUI. In this work, we consider three traffic patterns, i.e., UDP, VoIP, HTTP.

6.6.3 Data preprocessing

Feature Generation

This additional step allows generating more representative features based on the original ones. This step aims at increasing the relationship between the available features. Relations between timestamps, features, and labels are in Figure 6.21. Based on the dataset logged from the Amarisoft Web GUI, we generate the following features:

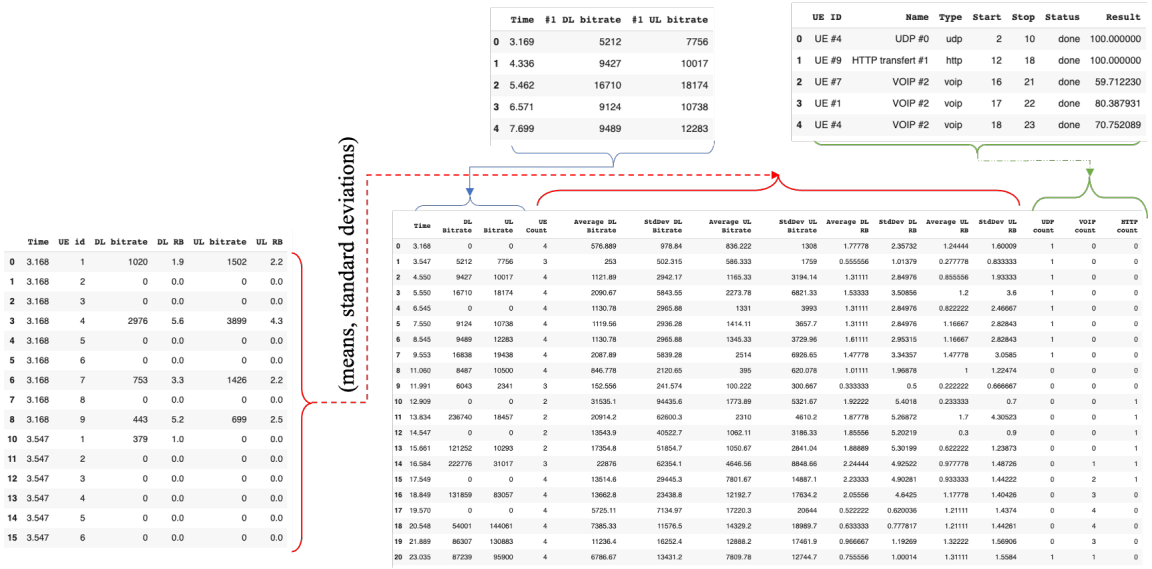
- Averages and standard deviations of downlink and uplink bitrate;
- Averages and standard deviations of downlink and uplink resource block;

To evaluate our model, we split dataset $\mathcal{D}$ into a training set and a testing set. The training set comprises 80% of the dataset, and it is used to obtain the model through the presented algorithms. The testing test is used to have a comparison with the prediction generated with the model.

Data Windowing

Before fitting an LSTM model to the dataset, we must transform the data. Specifically, the organization of data into input and output patterns where the observation at the previous time steps is used as an input to forecast the observation at the current time timestep.

Figure 6.22 illustrates the data windowing processing. The sequence $\mathbf{X}$ is split and grouped using a window W. Recall that by considering W steps before time t, we can classify how many traffic patterns appear at the considering timestamp t.



	Time	UE id	DL bitrate	DL RB	UL bitrate	UL RB
0	3.168	1	1020	1.9	1502	2.2
1	3.168	2	0	0.0	0	0.0
2	3.168	3	0	0.0	0	0.0
3	3.168	4	2976	5.6	3899	4.3
4	3.168	5	0	0.0	0	0.0
5	3.168	6	0	0.0	0	0.0
6	3.168	7	753	3.3	1426	2.2
7	3.168	8	0	0.0	0	0.0
8	3.168	9	443	5.2	699	2.5
10	3.547	1	379	1.0	0	0.0
11	3.547	2	0	0.0	0	0.0
12	3.547	3	0	0.0	0	0.0
13	3.547	4	0	0.0	0	0.0
14	3.547	5	0	0.0	0	0.0
15	3.547	6	0	0.0	0	0.0

	Time	#1 DL bitrate	#1 UL bitrate
0	3.169	5212	7756
1	4.336	9427	10017
2	5.462	16710	18174
3	6.571	9124	10738
4	7.699	9489	12283

	UE ID	Name	Type	Start	Stop	Status	Result
0	UE #4	UDP #0	udp	2	10	done	100.000000
1	UE #9	HTTP transfert #1	http	12	18	done	100.000000
2	UE #7	VOIP #2	voip	16	21	done	59.712230
3	UE #1	VOIP #2	voip	17	22	done	80.387931
4	UE #4	VOIP #2	voip	18	23	done	70.752089

	Time	DL Bitrate	UL Bitrate	UE Count	Average DL Bitrate	StdDev DL Bitrate	Average UL Bitrate	StdDev UL Bitrate	Average DL RB	StdDev DL RB	Average UL RB	StdDev UL RB	UDP count	VOIP count	HTTP count
0	3.168	0	0	4	576.889	978.84	836.222	1308	1.77778	2.35732	1.24444	1.60009	1	0	0
1	3.547	5212	7756	3	253	502.315	586.333	1759	0.555556	1.01379	0.277778	0.833333	1	0	0
2	4.550	9427	10017	4	1121.89	2942.17	1165.33	3194.14	1.31111	2.84976	0.855556	1.93333	1	0	0
3	5.550	16710	18174	4	2090.67	5843.55	2273.78	6821.33	1.53333	3.50856	1.2	3.6	1	0	0
4	6.545	0	0	4	1130.78	2965.88	1331	3993	1.31111	2.84976	0.822222	2.46667	1	0	0
5	7.550	9124	10738	4	1119.56	2936.28	1414.11	3657.7	1.31111	2.84976	1.16667	2.82843	1	0	0
6	8.545	9489	12283	4	1130.78	2965.88	1345.33	3729.96	1.61111	2.95315	1.16667	2.82843	1	0	0
7	9.553	16838	19438	4	2087.89	5639.28	2514	6926.65	1.47778	3.34357	1.47778	3.0585	1	0	0
8	11.060	8487	10500	4	846.778	2120.65	395	620.078	1.01111	1.96878	1	1.22474	0	0	0
9	11.991	6043	2341	3	152.556	241.574	100.222	300.667	0.333333	0.5	0.222222	0.666667	0	0	0
10	12.909	0	0	2	31535.1	94435.6	1773.89	5321.67	1.92222	5.4018	0.233333	0.7	0	0	1
11	13.834	236740	18457	2	20914.2	62600.3	2310	4610.2	1.87778	5.26872	1.7	4.30523	0	0	1
12	14.547	0	0	2	13543.9	40522.7	1062.11	3186.33	1.85556	5.20219	0.3	0.9	0	0	1
13	15.661	121252	10293	2	17354.8	51854.7	1050.67	2841.04	1.88889	5.30199	0.622222	1.23873	0	0	1
14	16.584	222776	31017	3	22876	62354.1	4646.56	8848.66	2.24444	4.92522	0.977778	1.48726	0	1	1
15	17.549	0	0	4	13514.6	29445.3	7801.67	14887.1	2.23333	4.90281	0.933333	1.44222	0	2	1
16	18.849	131859	83057	4	13662.8	23438.8	12192.7	17634.2	2.05556	4.6425	1.17778	1.40426	0	3	0
17	19.570	0	0	4	5725.11	7134.97	17220.3	20644	0.522222	0.620036	1.21111	1.4374	0	4	0
18	20.548	54001	144061	4	7385.33	11576.5	14329.2	18989.7	0.633333	0.777817	1.21111	1.44261	0	4	0
19	21.889	86307	130883	4	11236.4	16252.4	12888.2	17461.9	0.966667	1.19269	1.32222	1.56906	0	3	0
20	23.035	87239	95900	4	6786.67	13431.2	7809.78	12744.7	0.755556	1.00014	1.31111	1.5584	1	1	0

Figure 6.21: Instance and label creation.

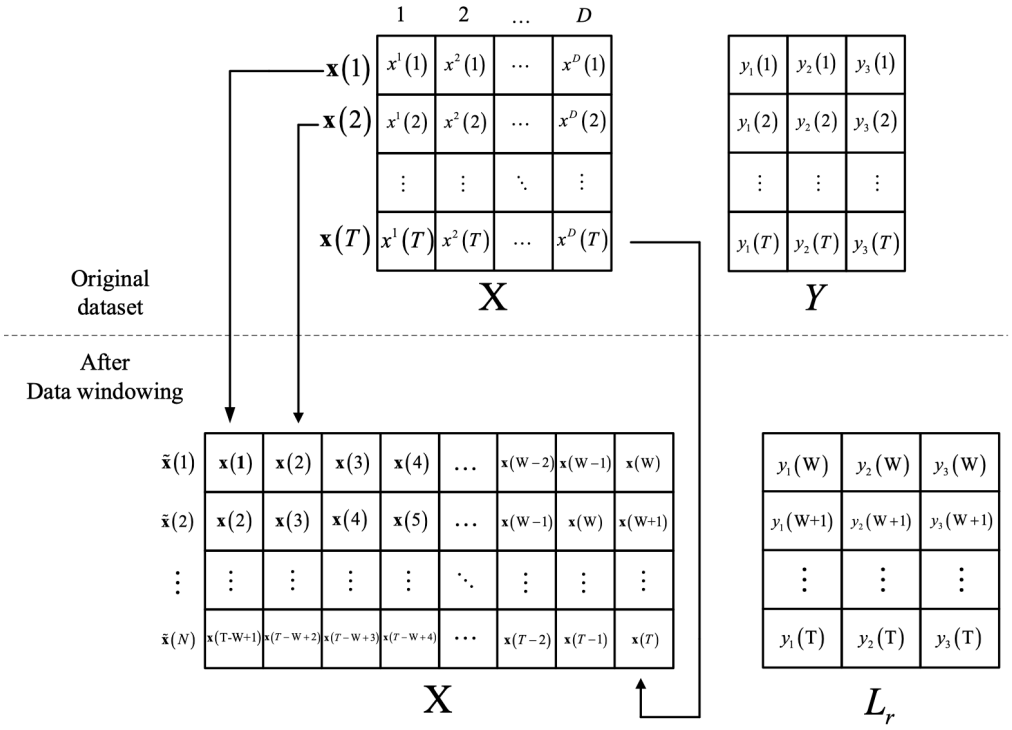


Figure 6.22: Dataset after data windowing

The vector $\mathbf{Y}$ is used to generate a new set of labels, namely $\mathbf{L_r}$, describing the traffic type associated to the last element in sequence $\tilde{\mathbf{x}}$. It means that by considering W steps before time t, we can recognize the traffic type at the considering instant t. The vector $\mathbf{L_r}$ has a dimension of $N \times L$.

Data Normalization

Sequence $\mathbf{X}$ is standardized by removing the mean and scaling to unit variance. This operation is performed to normalize the original sequence to reduce the variance of the input dataset.

The good practice usage with scaling techniques is as follows:

- Fit the scaler using available training data: this means the training data will be used to estimate the minimum and maximum observable values. This is done by calling the *fit()* function in *keras*.
- Apply the scale to training data: this means one can use the normalized data to train the model. This is done by calling the *transform()* function.
- Apply the scale to data going forward: this means one can prepare new data in the future on which they want to make predictions.

Moreover, the transform can be inverted for converting predictions back into their original scale for reporting or plotting.

6.6.4 Multi-class multi-output traffic pattern classifier

The simplest way to approach a multi-class multi-output problem is to divide it into multiple independent classification problems (one per class). Nonetheless, relationships between the labels are not considered, which can cause inconsistencies. To overcome those issues, we propose using LSTM networks.

Recall that LSTM networks are a particular type of RNN and are capable of tracking long-term dependencies into the input time series while solving the vanishing-gradient problem that affects standard RNNs. The capability to learn long-term dependencies is due to the structure of the LSTM cells, which incorporates gates regulating the learning process. The neurons in the hidden layers of an LSTM have the ability to retain or forget information about past input values by using structures called gates, which consist of a cascade of a neuron with a sigmoidal activation function and a point-wise multiplication block. In this way, the output of each memory cell possibly depends on the entire sequence of past states, making LSTMs suitable for processing time series with long time dependencies. The structure of our multi-class multi-output traffic pattern classifier is presented in Figure 6.23.

The first LSTM layer captures a new set of representations of input features. The rectifier linear unit (ReLu) activation function is used in the hidden layer unit.

To reduce the overfitting effect, which is a major problem if the amount of data in the training set is relatively small, a dropout layer is inserted after LSTM layer in the model at a 0.5 dropout rate. The representations after first LSTM hidden layer are fed with dropout ratio to regulate the learning.

Finally, the representations are fed to three fully connected layers with a number of each output equal to one and with linear activation to estimate target class values for each recognized label. In each output layer, the Linear activation function is used for the number of classes in each label estimation.

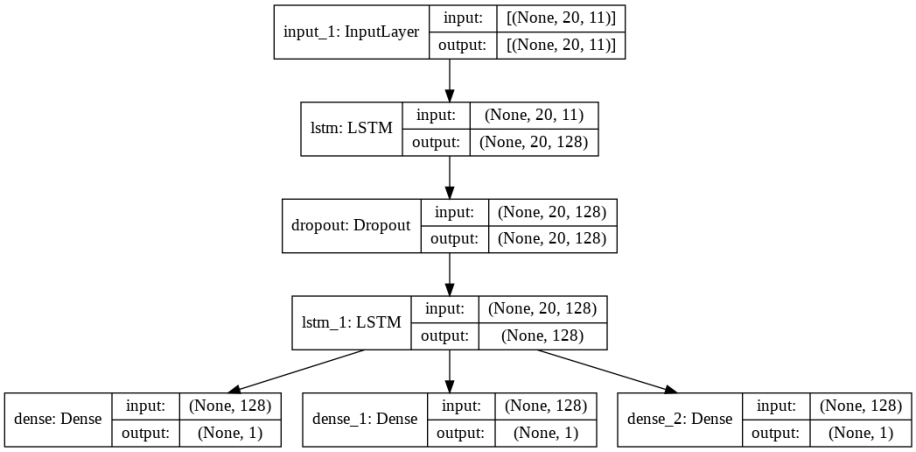


Figure 6.23: The multi-class multi-output traffic pattern classifier

Mean-square error (MSE) is a more suitable option for our model, which performs regression rather than classification because the estimated count values for each label are required to be non-negative integer values. Therefore, a class value estimation loss is calculated by MSE, whose formula is given as in (6.17).

In our experiment, we use the Adam optimizer for optimization as it is a popular technique and widely used successfully in other studies. The model is testing on the Testing Data to determine the effectiveness of the model in terms of evaluation metrics which are defined later in the next section.

6.6.5 Evaluation Metrics

Unlike single-label data, multi-label multi-class data may have different degrees of how many labels are contained within them. The two metrics for determining this are label names and cardinality of the label in a label set. The first metric is the label that the traffic belongs to. The cardinality of the label is the number of traffic patterns in the instances set. Data sets with the same label's names but different cardinality of label may behave differently.

As mentioned above, in single-label classification, we use simple metrics such as accuracy, precision, recall, etc. Unfortunately, at present, no public metric supports the multi-label multi-class classification task. In multi-label classification, a miss-classification is no longer a hard wrong or right. A classification result containing a subset of the actual labels should be considered better than it contains none of them, i.e., classifying two of three labels correctly is better than classifying no label at all. Given a set $\mathcal{D} = \{(\mathbf{x}_1, Y_1), ..., (\mathbf{x}_m, Y_m)\}$ of m test examples, the evaluation metrics of multi-label learning are calculated based on the average difference of the actual and the predicted set of labels overall test examples.

We can use the evaluation metrics of multi-class classification for each of the labels. In addition, we present a strict metric that indicates the exact match ratio between the actual and the predicted values. The details of these evaluation metrics are described below.

Hamming-Loss

Hamming loss reports how many times, on average, the relevance of an example to a class label is incorrectly predicted. Therefore, hamming loss takes into account the prediction error (an incorrect cardinality of the label is predicted) and missing error (a relevant cardinality of the label not predicted), normalized over the total number of label and the total number of examples.

$$\mathcal{H}Loss = \frac{1}{mL} \sum_{i=1}^{L} \sum_{j=1}^{m} \mathcal{I}\left(y_i(j) \neq \tilde{y}_i(j)\right), \tag{6.21}$$

where $\mathcal{I}$ is the indicator function. Ideally, we expect the hamming loss to be 0, which indicates no error; practically, the smaller the value of the hamming loss, the better the performance of the learning algorithm.

Exact Match Ratio - Subset Accuracy

It is the most strict metric, indicating the percentage of samples that have all their class of labels classified correctly.

$$SubsetAccuracy = \frac{1}{m} \sum_{i=1}^{m} I(Y_i = \tilde{Y}_i) \tag{6.22}$$

Unlike the multi-class classification problems, where the predicted labels have a chance of being partially correct, with the exact match ratio, all partially correct labels are ignored.

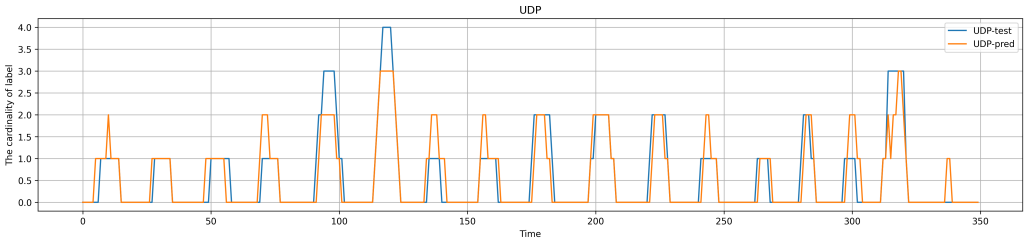
6.6.6 Performance Evaluation

In Figure 6.24, we present three plots (one for each traffic pattern) that show the actual cardinality of each label and the correspondent recognition over the test dataset. We can see that the classifier is capable of accurately identifying the labels contained in the label set. However, there are still errors in identifying the number class of labels. As can be seen from Table 6.1, UDP traffic recognition has the worst accuracy compared to VoIP and HTTP, with accuracy scores of 0.8, 0.83, and 0.93, respectively. The classifier mostly recognizes only the maximum number of labels of 3 for all types of labels while the true value is 4.

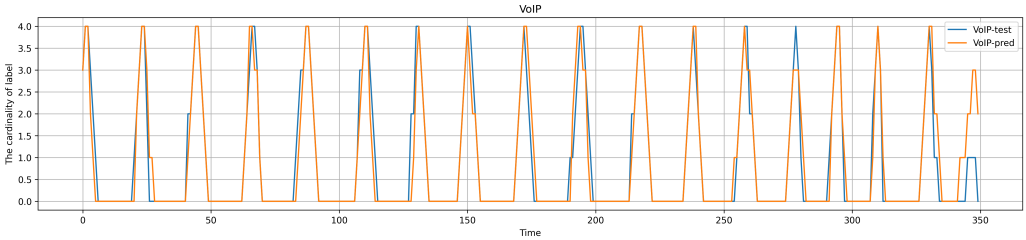
Moreover, the confusion matrices are presented in Table 6.2. The diagonal elements represent the number for which the predicted label cardinality is equal to the true value, while off-diagonal elements are those that are mislabeled by the classifier. We realize that most of the label values are correctly classified; the errors occur only at consecutive adjacent steps, mainly from UDP traffic.

The Hamming loss metric obtained by the proposed method on the testing set is $\mathcal{H}Loss = 0.13238$. This value as close to 0 indicates that the proposed classifier is good. Moreover, the proposed classifier achieves 0.678 the subset accuracy, which means 67.8% of label sets are correctly recognized/predicted. This is not an acceptable value because the rest percentage of subsets may have a chance of being partially correct, but with this exact match metric, we ignore those partially correct matches.

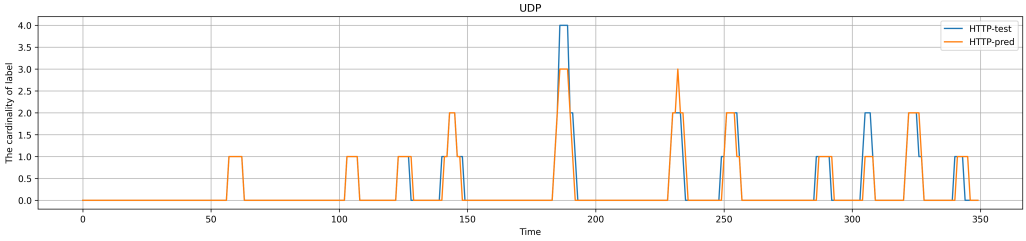
For the sake of completeness, we present in Table 6.3 the comparison between the proposed deep learning algorithm and two standard classifiers:



(a) Visualization of UDP traffic pattern.



(b) Visualization of VoIP traffic pattern.



(c) Visualization of HTTP traffic pattern.

Figure 6.24: Classification results on the test dataset of the three different labels

Traffic type	Accuracy Score	Precision score	
UDP	0.8171	0.8306	004
VoIP	0.8371	0.8315	248
HTTP	0.9314	0.9351	008

Table 6.1: Evaluation metrics of proposed classifier.

Confusion Matrix of		
UDP	VoIP	HTTP
$\begin{bmatrix} 221 & 6 & 0 & 0 & 0 \\ 16 & 54 & 8 & 0 & 0 \\ 3 & 9 & 15 & 0 & 0 \\ 0 & 2 & 6 & 6 & 0 \\ 0 & 0 & 0 & 4 & 0 \end{bmatrix}$	$\begin{bmatrix} 210 & 10 & 4 & 1 & 0 \\ 4 & 16 & 8 & 4 & 0 \\ 1 & 6 & 18 & 5 & 2 \\ 0 & 0 & 4 & 17 & 11 \\ 0 & 0 & 0 & 2 & 27 \end{bmatrix}$	$\begin{bmatrix} 275 & 5 & 0 & 0 & 0 \\ 6 & 36 & 2 & 0 & 0 \\ 0 & 6 & 16 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 \end{bmatrix}$

Table 6.2: Confusion matrices of each traffic pattern.

Algorithm	Subset Accuracy	Hamming Loss	ABC
LSTM	0.678	0.1276	004
LRMO	0.648	0.1252	248
KNMO	0.650	0.1144	008

Table 6.3: Results on different algorithms.

- Logistic Regression combined with Multioutput Classifier (LRMO);
- K-Neighbors combined with Multioutput Classifier (KNMO);

Adding any multi-label classification to a multi-output classifier allows multiple target variable classification. This strategy consists of fitting one classifier per target. The purpose of these approaches is to extend estimators to be able to estimate a series of target functions that are trained on a single predictor matrix to predict a series of responses. We can observe that although the Hamming loss of the proposed algorithm is higher than LRMO and KNMO algorithms, its subset accuracy achieves a better performance.

6.7 Summary

In the presented work, we have presented the models that allow highly accurate classification of traffic patterns from downlink control channel information, and without having to decode dedicated physical layer channels. Through DCI data, it is possible to extract features that allow reliable classification and prediction of the services that are being executed at the user equipment. For the identification and prediction of such traffic patterns, we have used Long Short-Term Memory (LSTM) network, which is imposed by the dataset characteristics since we use multivariate traffic information that derives directly from the DCI of the control channel. The LSTM units succeed in capturing the temporal correlation of the traffic even for distant timeslots. We have considered two scenarios of the multi-class traffic pattern classification/prediction and the multi-class multi-output traffic pattern classification. The labeled dataset of DCI data has been used to train and compare the classifiers.

In the multi-class scenario, the analysis has shown that by using an LSTM network, we can obtain the utmost results with more than 98% accuracy in recognition task and less than 10^1 loss metric in prediction task. Moreover, in the prediction task, a comparison with respect to using an autoencoder and without an autoencoder has also demonstrated that the prediction model with the autoencoder always offers a performance guarantee higher than the simple learning approach.

In the multi-class multi-output scenario, the proposed model is capable of accurately classifying the labels contained in the label set, although there are still errors in the prediction of the cardinality of each label. The performance of the model is evaluated by 67.8% in the most strict metric. In addition, the proposed models have been compared with the existing classification and regression models in the experiments, and the results illustrate the achieved improvements.

For future work, we plan to explore the applications of reinforcement learning techniques to derive the traffic pattern distribution and have better performance.

6.8 Resulting research contribution

The research contribution resulting from the work done in this chapter is below:

- T. A. Nguyen, P. Martins, "Cellular Traffic Type Recognition and Prediction," IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), 2021. *(submitted)*

Chapter 7

Conclusions

7.1 Summary of Achievements

In this thesis, we leverage the potential and capabilities of stochastic geometry and ML algorithms to analyze the cellular IoT network's performance, particularly the NB-IoT network. The thesis is organized into one preliminary part and three technical parts:

- in Chapters 1,2, and 3, we introduce the context and the problem. Moreover, we present related works on dimensioning networks based on both stochastic geometry and machine learning methods;
- in Chapter 4, we present an analysis of coverage probability;
- in Chapter 5, we focus on the dimensioning in the downlink of a multi-cell NB-IoT;
- in Chapter 6, we perform a classification and prediction of traffic patterns using LSTM networks.

The main contributions of this thesis on each specific technical part are:

1. Chapter 4 has been devoted to describing the coverage analysis in NB-IoT networks. In the first part of the work, we presented an analytical expression for the coverage and success access probability based on the stochastic geometry approach. The coverage probability was defined as the probability as a given device can achieve some thresholds of SINR. We showed that the implementation of the repetition transmission feature of NB-IoT could improve the access success performance. Then, we leveraged the machine learning algorithm to predict the coverage probability in a multi-cell network. We proved that each coverage probability of a network realization could be performed by the sigmoid-like function, and by using the neural network, we could estimate the coverage probability of a random NB-IoT network.
2. In Chapter 5, we performed an analysis to determine the number of required PRBs as a function of sensor user intensity and eNB intensity. We considered a multi-cell NB-IoT downlink scenario, where sensor nodes and eNBs were distributed according to Poisson point processes. Moreover, we presented a relationship between the required resources and the sensor user intensity given a target Quality of Service.

3. In Chapter 6, we presented the traffic pattern recognition (classification) and prediction problems. First, we considered a simple case, where there was only multi-class of traffic. Then, we extended our method to multi-class multi-output scenario and we performed the classification using LSTM networks. The results showed that by adapting deep-learning structures with LSTM, we could obtain the utmost results with more than 98% accuracy in recognition task and less than 10^{-1} loss metric in prediction task. In addition, the proposed models had been compared with the state-of-the-art algorithms, and the results showed the achieved improvements.

7.2 Future Works

In this thesis, we have studied a limited part of the problems from studying the NB-IoT network. However, due to the myriad of aspects in this topic, there are open issues relevant to defining additional future works.

As mentioned in the beginning, there are many applications in the NB-IoT network. One of them is applications related to traffic monitoring, smart metering. That leads to expanding the work of the thesis by applying different distributions to describe the various scenarios of the NB-IoT, such as Beta-Giniber point process or Cox point process driven by Poisson Line Process (to represent the devices distributed on roads in the applications such as an autonomous vehicle or Cooperative Intelligent Transport System).

In addition, the concept of repeated transmission is one of the advantages of the NB-IoT network to enhance the coverage. However, it may cause spectrum wastage. Motivated by this research gap, we can leverage the cognitive radio technique into the conventional NB-IoT radio resource allocation procedure.

Another possible direction to extend the work of this thesis is Reinforcement Learning (RL). RL is used to obtain optimal controls in the design of the system. In NB-IoT, there is a significant number of connected devices, and any factors from the channel may impact performance enhancement. Therefore, by leveraging RL, we can utilize feedbacks from the user side to avoid interference and complexity when learning the cost of the various task types.

Bibliography

[1] Cisco. Annual internet report (2018–2023) - white paper, March 2020. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html.

[2] R. Ratasuk, N. Mangalvedhe, Y. Zhang, M. Robert, and J. Koskinen. Overview of narrowband iot in lte rel-13. In *2016 IEEE Conference on Standards for Communications and Networking (CSCN)*, pages 1–7, 2016.

[3] Amarisoft. https://www.amarisoft.com/.

[4] H. Malik, H. Pervaiz, M. Mahtab Alam, Y. Le Moullec, A. Kuusik, and M. Ali Imran. Radio resource management scheme in nb-iot systems. *IEEE Access*, 6:15051–15064, 2018.

[5] M. Lebourges F. Baccelli, M. Klein and S. Zuyev. Stochastic geometry and architecture of communication networks, 1997.

[6] Jeffrey G. Andrews, Francois Baccelli, and Radha Krishna Ganti. A tractable approach to coverage and rate in cellular networks. *IEEE Transactions on Communications*, 59(11):3122–3134, 2011.

[7] Yingzhe Li, François Baccelli, Harpreet S. Dhillon, and Jeffrey G. Andrews. Statistical modeling and probabilistic analysis of cellular networks with determinantal point processes. *IEEE Transactions on Communications*, 63(9):3405–3422, 2015.

[8] Martin Haenggi. Mean interference in hard-core wireless networks. *IEEE Communications Letters*, 15(8):792–794, 2011.

[9] Overall description; Stage 2 (Release 13). *3GPP TS 36.300*, 13.4.0, June 2016.

[10] J. Xu, J. Yao, L. Wang, Z. Ming, K. Wu, and L. Chen. Narrowband internet of things: Evolutions, technologies, and open issues. *IEEE Internet of Things Journal*, 5(3):1449–1462, 2018.

[11] E. Berthelsen and J. Morrish. Forecasting the internet of things revenue opportunity. *Machina Res.*, 2015.

[12] Y. Wei R. S. Sinha and S. H. Hwang. A survey on lpwa technology: Lora and nb-iot. *ICT Exp.*, 3(1):14–21, 2017.

[13] Y. D. Beyene, R. Jantti, K. Ruttik, and S. Iraji. On the performance of narrow-band internet of things (nb-iot). In *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, 2017.

[14] Y. D. Beyene, R. Jantti, O. Tirkkonen, K. Ruttik, S. Iraji, A. Larmo, T. Tirronen, and a. J. Torsner. Nb-iot technology overview and experience from cloud-ran implementation. *IEEE Wireless Communications*, 24(3):26–32, 2017.

[15] U. Raza, P. Kulkarni, and M. Sooriyabandara. Low power wide area networks: An overview. *IEEE Communications Surveys Tutorials*, 19(2):855–873, 2017.

[16] Semtech Application Note. Lora modulation basics. AN1200.22, May 2015.

[17] A. Barto R. Sutton. *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press, 1998.

[18] S Bubeck and N Cesa-Bianchi. Regret analysis of stochastic and non-stochastic multi-armed bandit problems. *Found. Trends Mach. Learn.*, 5(1):1–122, 2012.

[19] A. Azari and C. Cavdar. Performance evaluation and optimization of lpwa iot networks: A stochastic geometry approach. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 206–212, 2018.

[20] M. Lauridsen, I. Z. Kovacs, P. Mogensen, M. Sorensen, and S. Holst. Coverage and capacity analysis of lte-m and nb-iot in a rural area. In *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, pages 1–5, 2016.

[21] Ansuman Adhikary, Xingqin Lin, and Y.-P. Eric Wang. Performance evaluation of nb-iot coverage. In *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, pages 1–5, 2016.

[22] Md Khalid Hossain Jewel Oluwole John Famoriji Fujiang Lin Md Sadek Ali, Yu Li. Channel Estimation and Peak-to-Average Power Ratio Analysis of Narrowband Internet of Things Uplink Systems. *Wireless Communications and Mobile Computing*, 2018.

[23] B. Vejlgaard, M. Lauridsen, H. Nguyen, I. Z. Kovacs, P. Mogensen, and M. Sorensen. Coverage and capacity analysis of sigfox, lora, gprs, and nb-iot. In *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, pages 1–5, 2017.

[24] M. Lauridsen, H. Nguyen, B. Vejlgaard, I. Z. Kovacs, P. Mogensen, and M. Sorensen. Coverage comparison of gprs, nb-iot, lora, and sigfox in a 7800 km^2 area. In *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, pages 1–5, 2017.

[25] M. Chafii, F. Bader, and J. Palicot. Enhancing coverage in narrow band-iot using machine learning. In *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, 2018.

[26] M. Haenggi, J. G. Andrews, F. Baccelli, O. Dousse, and M. Franceschetti. Stochastic geometry and random graphs for the analysis and design of wireless network. *IEEE Journal on Selected Areas in Communications*, pages 1029–1046, 2009.

[27] B. Blaszczyszyn F. Baccelli and P. Muhlethaler. Stochastic Analysis of Spatial and Opportunistic ALOHA. volume 27, page 1105–1119, Sept. 2009.

[28] X. Zhang and M. Haenggi. Random Power Control in Poisson Networks. volume 60, page 2602–2611, Sept. 2012.

[29] M. Haenggi. On Distances in Uniformly Random Networks. volume 51, page 3584–3586, Oct. 2005.

[30] H. S. Dhillon T. D. Novlan and J. G. Andrews. Analytical Modeling of Uplink Cellular Networks. volume 12, pages 2669–2679, June 2013.

[31] G. Gomez F. Javier Martin-Vega, F. Javier Lopez-Mart and M. Carmen Aguayo-Torres. Multi-User Coverage Probability of Uplink Cellular Systems: a Stochastic Geometry Approach. 2014.

[32] Jeffrey G. Andrews, Abhishek K. Gupta, and Harpreet S. Dhillon. A Primer on Cellular Network Analysis Using Stochastic Geometry. *arXiv e-prints*, 2016.

[33] Harpreet S. Dhillon Thomas D. Novlan and Jeffrey G. Andrews. Analytical Modeling of Uplink Cellular Networks. volume 12, pages 2669–2679, June 2013.

[34] K. S. Kim W. Huang, J. Choi. On the Impact of Channel Outage on the throughput of Fast Retrial Random Access for OFDMA Uplink. pages 1940–1943, 2008.

[35] Radha Krishna Ganti Martin Haenggi. *Interference in large wireless networks*. Now Foundations and Trends, 2009.

[36] Ansuman Adhikary Y. P. Eric Wang, Xingqin Lin. A Primer on 3GPP Narrowband Internet of Things (NB-IoT). volume 55, pages 117–123, March 2017.

[37] D. Raddino J. Schlienz. Narrowband Internet of things, Whitepaper. *Rohde-Schwarz*, 2016.

[38] A. Adhikary, X. Lin, and Y. . E. Wang. Performance evaluation of nb-iot coverage. In *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, pages 1–5, 2016.

[39] Further advancements for e-utran. *3GPP TR 36.814*, 1.5.2, Dec 2009.

[40] L. Bottou. *Online algorithms and stochastic approximations*. Cambridge University Press, Cambridge, UK., 1998.

[41] M. Haenggi and R. K. Ganti. *Interference in Large Wireless Networks (Foundations and Trends in Networking)*, volume 3. Hanover, MA, USA, Nov. 2008.

[42] M.S. Alouini H. ElSawy, A. Sultan-Salem and M. Z. Win. Modeling and analysis of cellular networks using stochastic geometry: A tutorial. volume 19, pages 167–203, 2017.

[43] R. Giuliano and F. Mazzenga. Dimensioning of ofdm/ofdma-based cellular networks using exponential effective sinr. *IEEE Transactions on Vehicular Technology*, 58(8):4204–4213, 2009.

[44] S. Doirieux M. Maqbool, M. Coupechoux and B.Baynat. Dimensioning methodology for ofdma networks. *In Proc. of the Wireless World Research Forum (WWRF22)*, 2009.

[45] B. Blaszczyszyn and M.K. Karray. Dimensioning of the downlink in ofdma cellular networks via an erlang's loss model. *In Proc. of European Wireless Conference*, 2009.

[46] M. K. Karray. Analytical evaluation of qos in the downlink of ofdma wireless cellular networks serving streaming and elastic traffic. *IEEE Transactions on Wireless Communications*, 9(5):1799–1807, 2010.

[47] B. Blaszczyszyn and M.K. Karray. Fading effect on the dynamic performance evaluation of ofdma cellular networks. *In Proc. of the 1st International Conference on Communications and Networking (ComNet)*, 2009.

[48] L. Decreusefond, E. Ferraz, P. Martins, and T. T. Vu. Robust methods for lte and wimax dimensioning. In *6th International ICST Conference on Performance Evaluation Methodologies and Tools*, pages 74–82, 2012.

[49] C. Yu, L. Yu, Y. Wu, Y. He, and Q. Lu. Uplink scheduling and link adaptation for narrowband internet of things systems. *IEEE Access*, 5:1724–1734, 2017.

[50] E. B. Rodrigues R. C. J. Neto and C. T. de Oliveira. Performance analysis of resource unit configurations for m2m traffic in the narrowband-iot system. *Proc. 35th Brazilian Commun. Signal Process. Symp.,*, page 816–820, 2017.

[51] S. Oh and J. Shin. An efficient small data transmission scheme in the 3gpp nb-iot system. *IEEE Communications Letters*, 21(3):660–663, 2017.

[52] L. Mroueh, Y. Yu, M. Terré, and P. Martins. Statistical uplink dimensioning in licensed cellular low power iot networks. In *2018 25th International Conference on Telecommunications (ICT)*, pages 527–531, 2018.

[53] S. Roman. The exponential polynomials and the bell polynomials, 4.1.3 and 4.1.8. *The umbral Calculus*, pages 63–67, 1984.

[54] M. Mihoubi. Bell polynomials and binomial type sequences. *Discrete Mathematics*, 308(12):2450–2459, 2008.

[55] H. D. Trinh, L. Giupponi, and P. Dini. Urban anomaly detection by processing mobile traffic traces with lstm neural networks. In *2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pages 1–8, 2019.

[56] Ke Wang, Xi Li, Hong Ji, and Xiaojiang Du. Modeling and optimizing the lte discontinuous reception mechanism under self-similar traffic. *IEEE Transactions on Vehicular Technology*, 65(7):5595–5610, 2016.

[57] Shahbaz Rezaei and Xin Liu. Deep learning for encrypted traffic classification: An overview. *IEEE Communications Magazine*, 57(5):76–81, 2019.

[58] N. Al Khater and R. E. Overill. Network traffic classification techniques and challenges. In *2015 Tenth International Conference on Digital Information Management (ICDIM)*, pages 43–48, 2015.

[59] Y. Choi, J. Y. Chung, B. Park, and J. W. Hong. Automated classifier generation for application-level mobile traffic identification. In *2012 IEEE Network Operations and Management Symposium*, pages 1075–1081, 2012.

[60] M. Finsterbusch, C. Richter, E. Rocha, J. Muller, and K. Hanssgen. A survey of payload-based traffic classification approaches. *IEEE Communications Surveys Tutorials*, 16(2):1135–1156, 2014.

[61] X. Han, Y. Zhou, L. Huang, L. Han, J. Hu, and J. Shi. Maximum entropy based ip-traffic classification in mobile communication networks. In *2012 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 2140–2145, 2012.

[62] Y. Fu, H. Xiong, X. Lu, J. Yang, and C. Chen. Service usage classification with encrypted internet traffic in mobile messaging apps. *IEEE Transactions on Mobile Computing*, 15(11):2851–2864, 2016.

[63] G Peter Zhang. Time series forecasting using a hybrid ARIMA and neural network model. *In: Neurocomputing*, 50:159–175, 2003.

[64] Wei-Chiang Hong. Application of seasonal SVR with chaotic immune algorithm in traffic flow forecasting. *In: Neural Computing and Applications*, 21.3:583–593, 2012.

[65] Huifang Feng and Yantai Shu. Study on network traffic prediction techniques. In *Proceedings. 2005 International Conference on Wireless Communications, Networking and Mobile Computing, 2005.*, volume 2, pages 1041–1044, 2005.

[66] C. Qiu, Y. Zhang, Z. Feng, P. Zhang, and S. Cui. Spatio-temporal wireless traffic prediction with recurrent neural network. *IEEE Wireless Communications Letters*, 7(4):554–557, 2018.

[67] Y. Zhao, Z. Zhou, X. Wang, T. Liu, Y. Liu, and Z. Yang. Celltrademap: Delineating trade areas for urban commercial districts with cellular networks. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, pages 937–945, 2019.

[68] Loutfi A L¨angkvist M, Karlsson L. A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognition Letters 42*, pages 11–24, 2014.

[69] Bengio Y Pascanu R, Mikolov T. Understanding the exploding gradient problem. *arXiv arXiv:1211.5063*, 2012.

[70] Bengio Y Pascanu R, Mikolov T. On the difficulty of training recurrent neural networks. *International Conference on Machine Learning*, 28:III–1310–III–1318, 2013.

[71] P. Frasconi Y. Bengio and P. Simard. The problem of learning long-term dependencies in recurrent networks. *IEEE International Conference on Neural Networks*, 3:1183–1188, 1993.

[72] Patrice Simard Yoshua Bengio and Paolo Fransconi. Learning Long-Term Dependencies with Gradient Descent is difficult. *IEEE transactions on Neural networks*, 5(2):157–166, March 1994.

[73] Yoshua Bengio Razvan Pascanu, Tomas Mikolov. On the difficulty of training recurrent neural networks. *Proceedings of the 30 th International Conference on Machine Learning*, 28(2):157–166, 2013.

[74] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Journal Neural Computation*, 9(8):1735–1780, Nov 1997.

[75] J. Berkowitz Z. C. Lipton and C. Elkan. A Critical Review of Recurrent Neural Networks for Sequence Learning. *Neural and Evolutionary Computing*, May 2015.

[76] T. Prabhu K. Jijo George, A. Sivabalan and Anand R. Prasad. End-to-end mobile communication security testbed using open source applications in virtual environment. *Journal of ICT Standardization*, 3(4):67–90, 2015.

[77] https://github.com/anhnt-telecomparis/Dataset/blob/main/Data7.csv.

[78] Cristopher M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[79] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014.

[80] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *In: arXiv preprint*, 2014.

Appendix A: Machine Learning Libraries and Implementation

Name	Type	Characteristics
Numpy	Python Library	Math library that adds support for large, multi-dimensional arrays and matrices. It includes a large collection of high-level mathematical functions to operate on arrays.
Scikit-Learn	Python Library	ML library that includes fundamental classification, regression and clustering algorithms. It is used for data preprocessing and for algorithms benchmark.
Pandas	Python Library	Library for data manipulation and analysis. It offers data structures and operations for manipulating numerical. It is used to load the data and for preprocessing.
Matplotlib	Python Library	Plotting library for NumPy. It provides an object-oriented API for embedding plots into applications. It is designed to closely resemble that of MATLAB.
Tensorflow	Python Library	End-to-end platform for ML dataflow and differentiable programming across a range of tasks. It has a comprehensive, flexible ecosystem of tools that lets researches push the state of the art in ML and developers build and deploy ML applications. It is used for both research and production at Google.
Keras	Python Library	Library for neural-network implementations. It is capable of running on top of TensorFlow, R and other ML backends. It is designed to enable fast experimentation with deep neural networks.
Google Colab	Dev Environment	Google's free cloud service for AI developers similar to Jupyter Notebook. It integrates main ML libraries and allows multi-user contributions. There are free GPU and TPU for neural networks training.

ECOLE
DOCTORALE

Titre : Dimensionnement d'un réseau cellulaire IoT en utilisant des techniques de géométrie stochastique et d'apprentissage automatique.

Mots clés : NB-IoT, dimensionnement, la géométrie stochastique, l'apprentissage automatique

Résumé : L'Internet des objets à bande étroite (NB-IoT) est une technologie de réseau étendu à faible consommation d'énergie, qui a été normalisée dans 3GPP, spécifie une nouvelle procédure d'accès aléatoire et un nouveau schéma de transmission pour l'IoT. Les avantages du réseau NB-IoT sont une couverture étendue, une faible consommation d'énergie et la prise en charge d'un grand nombre de connexions. En particulier, le réseau NB-IoT peut connecter efficacement jusqu'à 50 000 dispositifs par cellule de réseau NB-IoT.

Nous concentrons notre travail sur l'étude du dimensionnement des réseaux NB-IoT. À cet égard, nous utilisons des techniques de géométrie stochastique et d'apprentissage automatique et cette thèse caractérise les indicateurs de performance clés du réseau NB-IoT, tels que la probabilité de couverture, le nombre de blocs de ressources radio nécessaires, ainsi que la reconnaissance et la prédiction des modèles de trafic sur la base des informations de contrôle en liaison descendante. La thèse est divisée en trois études principales.

Premièrement, nous dérivons les performances de la probabilité de couverture de la liaison montante dans un réseau NB-IoT à cellule unique et à cellules multiples. Les expressions analytiques de la couverture et des probabilités d'accès réussi dans un réseau NB-IoT monocellulaire sont présentées en considérant la distribution d'arrivée des paquets. Dans le scénario multi-cellules, une prédiction de la probabilité de couverture est déterminée directement à partir des paramètres du réseau en utilisant un réseau neuronal profond. L'analyse suivante consiste en un modèle analytique permettant de calculer les blocs de ressources radio nécessaires dans le réseau NB-IoT multi-cellules et de déterminer la probabilité de panne du réseau. Ce modèle est bénéfique pour les opérateurs car il clarifie la façon dont ils doivent gérer le spectre disponible. Enfin, la thèse aborde les problèmes de reconnaissance et de prédiction du type de trafic en utilisant les données collectées à partir des informations de contrôle de la liaison descendante. Un large groupe d'algorithmes d'apprentissage automatique est mis en œuvre et comparé pour identifier celui avec les meilleures performances.

L'analyse menée dans cette thèse démontre que la géométrie stochastique et les techniques d'apprentissage automatique peuvent servir d'outils puissants pour analyser les performances du réseau NB-IoT. Les framewoks développés dans ce travail fournissent des outils analytiques généraux qui peuvent être facilement étendus pour faciliter d'autres recherches sur les réseaux 5G.

Title : Dimensioning Cellular IoT network using Stochastic geometry and Machine learning techniques

Keywords : NB-IoT, dimensioning, stochastic geometry, machine learning

Abstract : Narrowband Internet of Things (NB-IoT) is a Low Power Wide Area technology, which was standardized in the Third Generation Partnership Project release, specifies a new random access procedure and a new transmission scheme for IoT. The advantages of the NB-IoT network are providing deep coverage, low power consumption, and support of a huge number of connections. Especially, NB-IoT can efficiently connect up to 50,000 devices per NB-IoT network cell.

We focus our work on the study of NB-IoT network dimensioning. In this regard, we use stochastic geometry and machine learning techniques along with the thesis to characterize key performance indicators of the NB-IoT network, such as coverage probability, the number of required radio resource blocks, and the traffic pattern recognition and prediction based on the downlink control information. The thesis is divided into three major studies.

Firstly, we derive the performance of uplink coverage probability in single-cell and multi-cell of NB-IoT network. The analytical expressions of the coverage and successful access probabilities in a single-cell NB-IoT network are presented by considering the packet arrival distribution. In the multi-cell scenario, a prediction of coverage probability is determined directly from the network parameters by using a Deep Neural Network. The subsequent analysis consists of an analytical model to calculate the required radio resource blocks in the multi-cell NB-IoT network and determine the network outage probability. This model is beneficial for operators because it clarifies how they should manage the available spectrum. Finally, the thesis addresses the recognition and prediction traffic type problems using the data collected from the Downlink Control Information. A wide group of machine learning algorithms are implemented and compared to identify the highest performances.

The analysis conducted in this thesis demonstrates that stochastic geometry and machine learning techniques can serve as powerful tools to analyze the performance of the NB-IoT network. The frameworks developed in this work provide general analytical tools that can be readily extended to facilitate other research in 5G networks.

Institut Polytechnique de Paris
91120 Palaiseau, France