Mémoire présenté devant l'ENSAE ParisTech
pour l'obtention du diplôme de la filière Actuariat
et l'admission à l'Institut des Actuaires
le 08/11/2017

Par : **Isaac Haik**

Titre : Text mining et reconnaissance d'écriture appliqués à l'assurance

Confidentialité : ☒ NON ☐ OUI (Durée : ☐ 1 an ☐ 2 ans)

Les signataires s'engagent à respecter la confidentialité indiquée ci-dessus

Membres présents du jury de la filière

Nom : Romuald Elie
Signature :

Entreprise : Milliman
Nom : Laurent Devineau
Signature :

Membres présents du jury de l'Institut des Actuaires

Directeur du mémoire en entreprise :
Nom : Antoine Ly
Signature :

Autorisation de publication et de mise en ligne sur un site de diffusion de documents actuariels
(après expiration de l'éventuel délai de confidentialité)

Signature du responsable entreprise

Secrétariat :

Signature du candidat

Bibliothèque :

École Nationale de la Statistique et de l'Administration Économique (ENSAE)
5, avenue Henry Le Chatelier - 91120 PALAISEAU, FRANCE

Remerciements

Je tiens à remercier le cabinet de conseil en actuariat Milliman pour la possibilité qui m'a été donné de travailler dans un cadre de travail agréable et motivant. Je remercie particulièrement toute l'équipe R&D de Milliman Paris pour les conversations enrichissantes et épanouissantes que nous avons tenues.

Plus particulièrement, je remercie Laurent Devineau, Directeur du département R&D, pour l'encadrement de qualité, sa pédagogie et son intérêt pour l'apprentissage statistique. Merci pour tout le temps et l'attention accordés à ce sujet de mémoire.

Je remercie Antoine Ly pour son encadrement tout au long de ce stage et le temps consacré à la problématique de ce mémoire. Ses conseils et ses idées auront été les moteurs de ce stage et de ce mémoire. Je remercie également Alexandre Boumezoued pour ses suggestions sur l'utilisation de mes algorithmes dans un cadre de constructions de tables de mortalité.

Je remercie les enseignants de l'ENSAE et du master Mathématiques Vision et Apprentissage de l'ENS Paris Saclay pour la qualité de leurs cours lors de cette année scolaire. Ils m'ont été d'une grande utilité pour la rédaction de ce mémoire et seront, à n'en pas douter d'une grande aide pour ma vie professionnelle.

Je remercie également mes camarades de l'ENSAE et du MVA. Leur collaboration tout au long de cette année intense aura été indispensable.

Enfin, je remercie mon frère, mes soeurs et mes parents pour leur soutien au quotidien et particulièrement mon père pour sa relecture de mon mémoire.

Résumé

La reconnaissance de documents dactylographiés et manuscrits est un défi pour les assureurs soucieux de fluidifier leur relation client. La reconnaissance du texte mais aussi son exploitation sont désormais possibles par l'émergence de modèles mathématiques qui composent l'intelligence artificielle.
La reconnaissance textuelle se compose d'une succession d'étapes. Si les similarités de lettres dactylographiées suffisent pour leur reconnaissance, l'écriture manuscrite propre à chacun, irrégulière et enchevêtrée nécessite davantage de modélisation. L'utilisation de réseaux de neurones, architecturés minutieusement, combinée à des bases d'entraînement de taille relativement moyenne obtient des résultats déjà prometteurs avec près de 80% des lettres manuscrites reconnues.
Le texte est une donnée difficilement interprétable pour un ordinateur et notamment dans le cadre du suivi de la réputation d'un assureur. Cette dernière peut être mesurée par un ensemble de nouvelles données disponibles telles que les réactions d'assurés sur différents sites spécialisés mais aussi sur les réseaux sociaux. Les modèles donnant du sens aux données textuelles, se basent sur une approche fréquentiste ou contextuelle permettant d'obtenir une représentation mathématiques du texte.
Une utilisation adéquate de l'intelligence artificielle peut répondre à des problématiques fortes présentes chez un assureur. La donnée textuelle n'est plus désormais une donnée que seul l'humain peut appréhender. Dans un avenir proche, une paramétrisation plus fine de ces modèles permettra leurs déploiements dans des conditions réelles.

Mots-clés : Intelligence artificielle, OCR, machine-learning, text-mining, contrat d'assurance, risque de réputation.

Abstract

For some years, insurers have been beginning a digital transformation of their services. This transformation, along with the emergence of new analysis and prediction technologies, questions the daily work of an actuary. Thus, tools and methods linked to artificial intelligence have to be appropriated by actuaries in order to control their risks.

Several actors of the insurance market have been already drawing benefit from this digital transformation by using new data provided for instance by telematics. Moreover, a lot of data are still unexploited. Indeed, it is not unusual that claim reports, expert reports or medical certificates are stored numerically. The unstructured nature of those documents (scanned) prevented them from being directly used. Exploitation of these documents is usually led manually by human analysis to help the agent in taking decisions (claims management, expertise,etc). The goal of this thesis is to introduce an automatic analysis method of documents.
Starting by slightly differentiating documents : typescript, handwritten, digital, we focus on the understanding of actual methods. Then, regarding the handwritten documents analysis, we noticed that research is still in progress on that field that the reason why we decided to dedicate one part on that issue. We indeed developed and trained our own convolutional neural network. Aware of the complexity of such model, we pay attention to detail the mechanism of those Deep Learning techniques.
Afterwards, we highlight benefits of such method to extracted semantic information from scanned documents. Methodologies to finally exploit text is introduced as a second main part of this thesis. We indeed present a concrete application of text mining.

Keywords : Artificial intelligence , OCR, machine-learning, Text mining, insurance contracts, reputation risks.

Note de synthèse

Contexte

Au début de l'année 2017, une compagnie d'assurance vie japonaise a annoncé le remplacement de trente-quatre de ses salariés chargés de l'étude des dossiers de souscription des clients par une intelligence artificielle. Ces méthodes ont la capacité de lire, de comprendre et d'analyser des documents relatifs à la souscription d'une personne à un contrat d'assurance vie afin de décider rapidement de la tarification à appliquer. Si le développement et la mise en service de ces outils sont pour l'instant externalisés, les assureurs vont être progressivement amenés à s'approprier ces outils et plus généralement les moyens de l'intelligence artificielle afin d'en maîtriser les risques, mais surtout d'être en adéquation avec les nouveaux besoins des assurés de plus en plus connectés.
Reconnaître et comprendre des documents dans le but d'automatiser des processus de décision contribuent à l'accélération et à l'amélioration de la souscription d'un contrat d'assurance. Deux types de documents sont à distinguer : les tapuscrits (écrits sur un ordinateur) assez bien traités par des algorithmes *open-source* toujours améliorables ; et les manuscrits. La reconnaissance de l'écriture manuscrite, beaucoup plus aléatoire, est encore l'objet de nombreuses recherches scientifiques. Dans ce mémoire, nous présentons une méthode qui s'appuie sur certaines publications récentes dans le domaine du *Deep Learning* et qui a obtenu des résultats prometteurs. Combinées à des algorithmes de reconnaissance de la mise en page et de la police, brièvement étudié dans ce mémoire, il est ainsi possible d'extraire une information textuelle d'un document sans avoir recours à un opérateur humain.
Dans un second temps, ce texte brut peut être analysé par des méthodes d'exploitation de la donnée textuelle, ce qui est communément appelé le *text mining*. Nous exposerons un ensemble de techniques permettant d'exploiter ces données non structurées au travers d'un exemple : l'analyse de la réputation d'un assureur.

Les nouveaux enjeux de la data science dans la reconnaissance de caractères et l'extraction d'informations

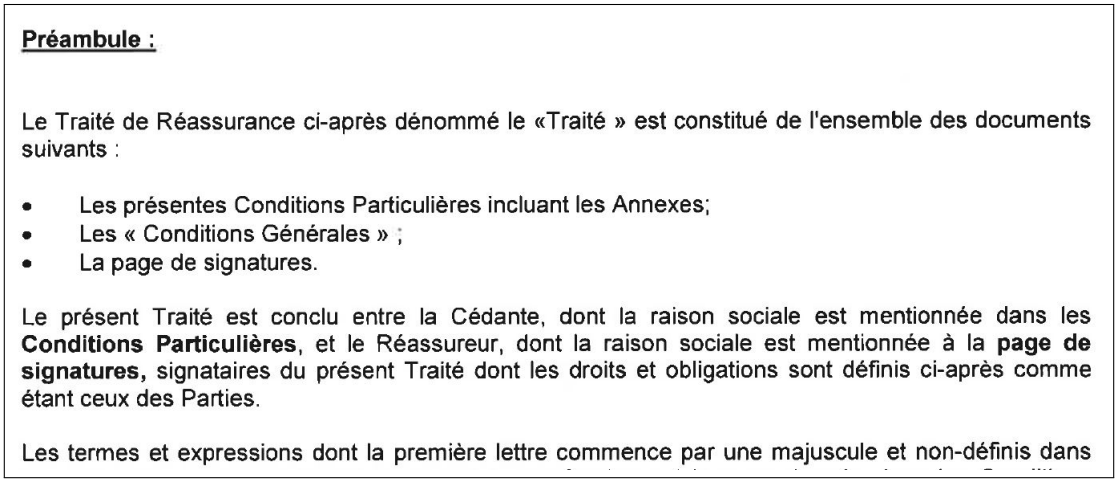
La data science de manière générale et le *Deep Learning* ont révolutionné les algorithmes pratiquant la *reconnaissance optique de caractères* (OCR). Certaines tâches comme la reconnaissance de lettres ou de chiffres ont vu leurs résultats s'améliorer nettement grâce, notamment, aux réseaux de neurones. Grâce à des architectures bien spécifiques (convolution, récurrence, etc.), il est possible de reconnaître des séquences de mots manuscrits. La fouille de texte, l'analyse de sentiments ou la classification de documents permettent alors de donner du sens à l'information extraite d'un document.

La reconnaissance de l'écriture tapuscrite

Les assureurs peuvent posséder tous types de contrats scannés non exploitables directement. Afin d'automatiser leur compréhension, il convient de reconnaître le texte tapuscrit.
Caractérisé par sa régularité (typologie normée), il est aujourd'hui exploitable par de nombreux algorithmes. Ces derniers comportent généralement plusieurs étapes : détection de la mise en page, segmentation du texte en composantes connectées [1],segmentation des mots et analyse des polices.
Certaines de ces méthodes sont libres d'accès et implémentées dans de nombreux modules. A des fins de compréhension, nous proposons de détailler les étapes d'un module particulier. Cette

1. fractions d'images où les pixels noirs se suivent

analyse nous aide notamment dans l'élaboration de notre propre modèle de reconnaissance d'écriture manuscrite.



Préambule :

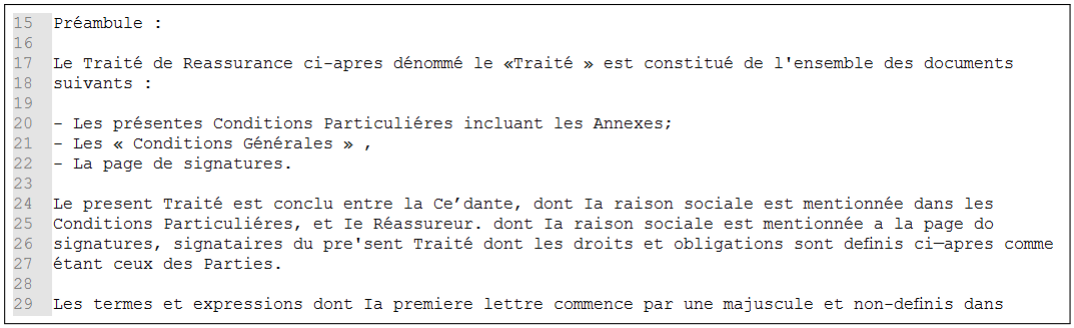
Le Traité de Réassurance ci-après dénommé le «Traité » est constitué de l'ensemble des documents suivants :

- Les présentes Conditions Particulières incluant les Annexes;
- Les « Conditions Générales » ;
- La page de signatures.

Le présent Traité est conclu entre la Cédante, dont la raison sociale est mentionnée dans les **Conditions Particulières**, et le Réassureur, dont la raison sociale est mentionnée à la **page de signatures,** signataires du présent Traité dont les droits et obligations sont définis ci-après comme étant ceux des Parties.

Les termes et expressions dont la première lettre commence par une majuscule et non-définis dans

(a) Image initiale



```
15  Préambule :
16
17  Le Traité de Reassurance ci-apres dénommé le «Traité » est constitué de l'ensemble des documents
18  suivants :
19
20  - Les présentes Conditions Particuliéres incluant les Annexes;
21  - Les « Conditions Générales » ,
22  - La page de signatures.
23
24  Le present Traité est conclu entre la Ce'dante, dont Ia raison sociale est mentionnée dans les
25  Conditions Particuliéres, et Ie Réassureur. dont Ia raison sociale est mentionnée a la page do
26  signatures, signataires du pre'sent Traité dont les droits et obligations sont definis ci—apres comme
27  étant ceux des Parties.
28
29  Les termes et expressions dont Ia premiere lettre commence par une majuscule et non-definis dans
```

(b) Texte reconnu

FIGURE 1 – Exemple d'analyse automatique d'un contrat de réassurance scanné

Détection d'écriture manuscrite et segmentation en ligne et en mots

Malgré leurs performances correctes sur des documents dactylographiés, ces méthodes ne fonctionnent pas sur des documents manuscrits. Or, si les échanges entre assureurs et assurés sont de plus en plus numérisés, les documents manuscrits ou partiellement manuscrits scannés et envoyés à l'assureur sont encore légion.
Dans certains documents, seulement une partie du document est constituée d'écriture manuscrite. Afin de détecter cette zone de texte et l'extraire, une analyse de l'épaisseur des caractères et des lignes permet de distinguer l'écriture manuscrite de celle dactylographiée. De manière générale, les traits issus de l'écriture manuscrite sont en effet plus épais et moins rectilignes.
Une fois ces zones délimitées, celles-ci sont segmentés en lignes et en mots par des modèles de mélanges gaussiens. Cela constitue la première étape de notre analyse de documents manuscrits.

(a) Image initiale

(b) Mots retrouvés

FIGURE 2 – Application d'un mélange gaussien pour la segmentation en mots

Reconnaissance de l'écriture par modèle de chaînes de Markov cachées (HMM)

Chaque image de mot peut être segmentée en graphèmes (portions de lettres). Différent d'une segmentation en lettres, les graphèmes permettent de contourner la difficulté posée par l'entrecroisement des caractères.
En fonction de leur forme, les graphèmes sont ensuite classés. Chaque mot est ainsi représenté par une succession de classes que nous tentons de modéliser au travers d'un modèle de markov caché (HMM). Le HMM est un modèle qui suppose qu'il existe une chaîne de Markov latente qui explique la séquence observée. Ainsi l'idée est d'être capable de modéliser la séquence chronologique des traits lors de l'écriture d'un mot.
Ces modèles ont été parmi les plus performants jusqu'au début des années 2000. Toutefois le Deep Learning a changé la façon de concevoir la vision par ordinateur et des modèles, plus adaptés à notre problème, ont émergé. En effet, la modélisation HMM ne tient pas compte du contexte général du mot. En effet, la prédiction du graphème suivant ne tient compte que du graphème présent.
D'autre part, ce modèle présente d'autres limites comme la segmentation en graphèmes, souvent délicate. Certaines approches du Deep Learning proposent aujourd'hui de nouvelles alternatives.

Reconnaissance de l'écriture par réseaux de neurones

Le Deep Learning ou *apprentissage profond* est devenu aujourd'hui la norme dans le traitement d'images. Popularisé par des résultats probants lors de concours de reconnaissance d'images (Imagnet), le Deep Learning s'est peu à peu imposé comme un élément majeur du traitement de données non structurées (images, textes, sons ou encore odeurs).

Dans le cadre de la reconnaissance de l'écriture manuscrite, il convient de définir l'architecture de notre modèle d'apprentissage profond. En effet, il existe une multitude de structures neuronales possibles permettant de répondre à notre problématique. Par exemple, le *feedforward network* est un réseau de neurones comparable à une suite de régressions linéaires. Il peut être entraîné, par exemple, sur des données structurées (exemple : les données d'un assuré, son âge, son sexe, sa garantie,etc.) et peut répondre conjointement aux questions : l'assuré est-il à risque ? quel est son niveau de risque ? etc. Les *convolutional neural networks* (CNN) sont, quant à eux, des réseaux mieux adaptés à l'analyse d'images. Ils sont constitués de couches de convolutions qui ont la possibilité d'extraire de l'information sur des matrices numériques. Enfin, les *recurrent neural networks* ont eux la possibilité de prendre en compte l'aspect séquentiel de données. Ce sont des réseaux qui ont des couches de neurones qui communiquent à l'intérieur d'elles mêmes prenant ainsi en compte la temporalité des données.

S'appuyant sur la littérature, nous nous inspirons de ces différentes structures afin de définir et d'entraîner notre propre modèle. Ainsi, le fonctionnement général de notre réseau peut se résumer ainsi : étant donnée l'image d'un mot, des points saillants sur l'image vont être détectés grâce aux couches de convolution puis la séquence de ces points est modélisée par des couches de neurones récurrentes. Finalement une couche de classification simple similaire à celle d'un feedforward network mais adaptée à notre problème (*CTC layer*) renvoie une matrice de probabilités. A partir de cette matrice, on distingue le mot reconnu. La figure suivante présente la structure complète du réseau que l'on entraîne :

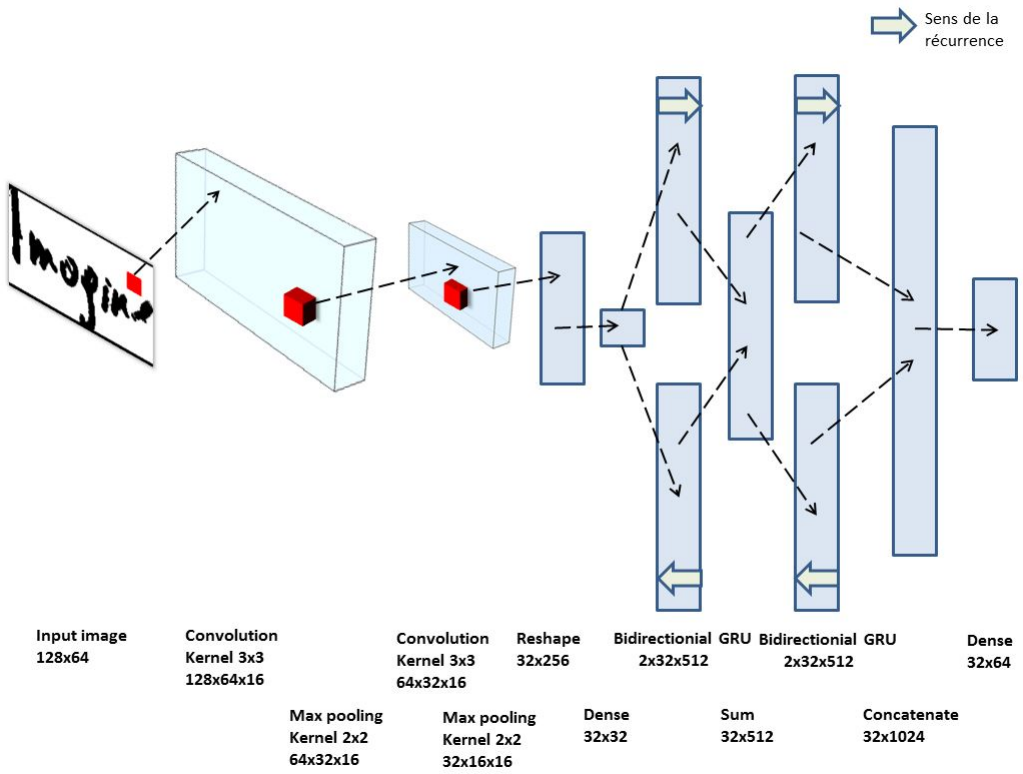


FIGURE 3 – Réseau de neurones utilisé

Notre réseau comprend 49 millions de paramètres à optimiser. Cette grandeur nous a alors amené à utiliser de grandes capacités de calcul et notamment une grande quantité de processeurs (GPU : Graphical Processor Units). Ce réseau est entraîné sur différentes bases de données libres d'accès composées de 50 000 à 100 000 images de mots. Voici un exemple de résultats :

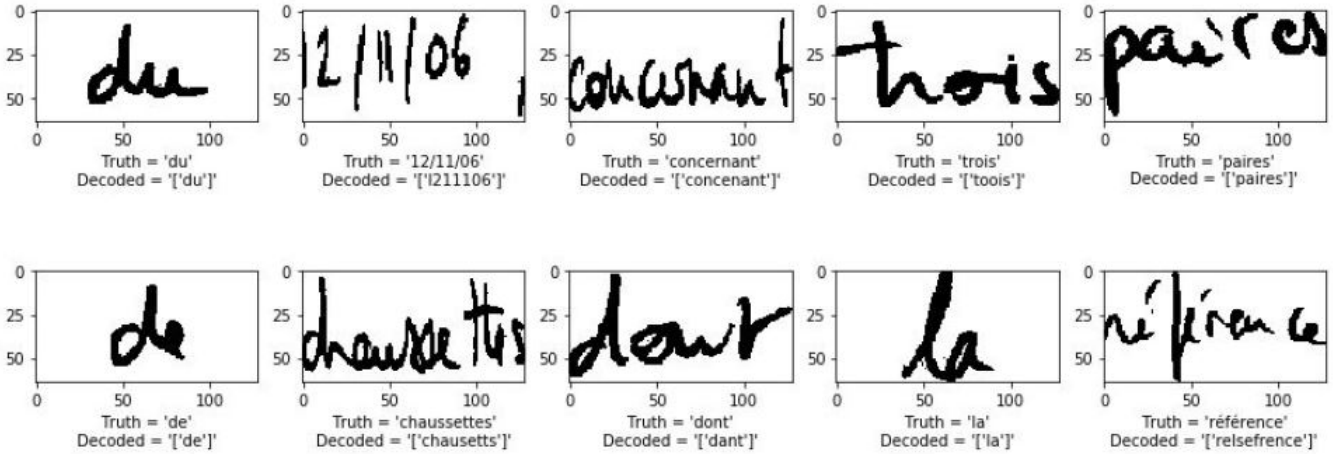


FIGURE 4 – Résultats de la reconnaissance de l'écriture manuscrite par les réseaux de neurones

Les résultats sont globalement satisfaisants sur une base de données de test similaire à la base de données d'entraînement. Près de 50% mots sont entièrement reconnus et un correcteur orthographique bien calibré augmente ce score en corrigeant la plupart des erreurs. Toutefois, sur des mots d'épaisseurs différentes ou sur des images scannées avec une faible qualité, les résultats se dégradent très rapidement. Cela est dû au fait que la base de données d'entraînement ne comporte pas assez d'exemples de types divers.

Utilisation de l'algorithme dans différents cadres

Même si en l'état, l'algorithme que nous avons entraîné ne peut être mis en production, les résultats obtenus sont prometteurs. En tenant compte de la diversité des typographies existantes et en renforçant l'entraînement de notre modèle, nous sommes confiants dans l'amélioration de ses performances.
Au delà de l'analyse automatique des constats automobiles, le champ d'application de cette méthode est vaste. Par exemple, la lecture automatique d'anciens certificats de décès à des fins d'études statistiques sur les causes de mortalité des années 60, l'étude automatique des actes de naissance pour une évaluation de l'exposition dans le calcul de tables de mortalité ou encore, la lecture de scans de contrats de réassurance pour la mutualisation de taux minimum garantis.

Analyse de l'information extraite

Une fois le texte d'un document reconnu, il convient d'en faire l'analyse automatique en utilisant du **text mining**. Les données de la première partie ne pouvant être directement utilisées, nous illustrons un ensemble de méthodes de text mining dans le cadre d'une analyse de la réputation d'un assureur à partir des commentaires postés sur un forum spécialisé. *opinion-assurances.fr*.

Le text mining est constitué d'un ensemble de méthodes permettant de constituer une représentation numérique des mots et d'en extraire de l'information. Ces méthodes se basent soit sur une approche fréquentiste tels que l'analyse *tf-idf* qui tend à donner un poids important aux mots rares mais fréquent au sein d'un document, soit sur une approche sémantique et syntaxique du texte. Deux mots vont avoir des représentations vectorielles similaires s'ils ont un sens proche. Dans le deuxième cas, ce sont des réseaux de neurones qui vont créer cette représentation. Par ailleurs, des graphes de mots peuvent permettre également une compréhension des relations entre ces derniers et une extraction de mots-clés.

Extraction de mots-clés suivant la satisfaction des assurés

La première analyse que l'on fait sur les commentaires des assurés consiste en une extraction des mots-clés afin de comprendre les principaux points que les assurés veulent mettre en avant. L'extraction des mots-clés peut se faire par une analyse sur le graphe de mots (analyse *k-core*), les noeuds du graphe sont les mots tandis que les arrêtes indiquent si les mots se suivent. Dans *l'analyse k-core*, un core est un sous-graphe où les noeuds ont un certain nombre de voisins. Une deuxième manière de faire est une analyse bayésienne de la représentation tf-idf de chaque commentaire. Le résultat en utilisant la première méthode, puis en prenant par exemple le sous-graphe centré autour du mot "aucun" est celui-ci :

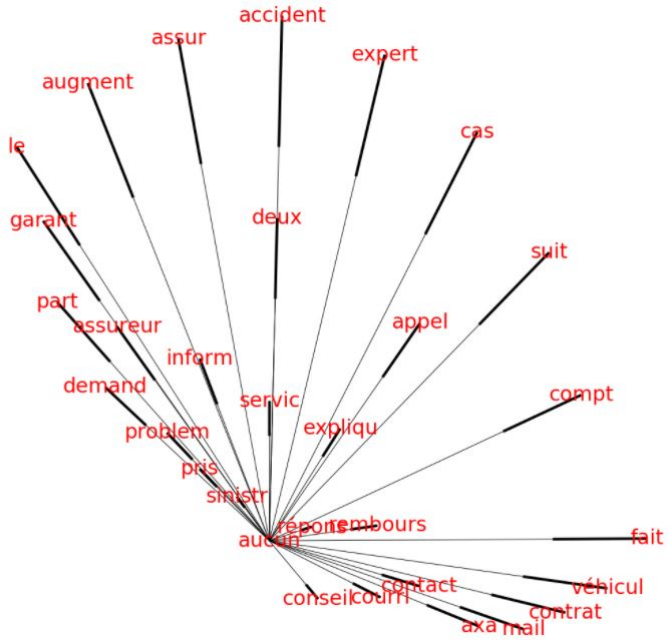


FIGURE 5 – Sous-graphe du principal core centré autour du mot 'aucun'

Le résultat en utilisant la représentation vectorielle *tf-idf* et en essayant d'identifier les *bigrams* (paires de mots) ou *trigrams* les plus importants donnent le résultat suivant

Niveau de Satisfaction	**1**	**2**	**3**	**4**	**5**
Orde d importance					
0	service client	déconseille vivement	bonne mais	rapport qualité	satisfaite agence
1	fuir absolument	très déçue	ancien intermédiaire	qualité prix	très bon
2	depuis ans	manque professionnalisme	commentaire moins	rapport qualité prix	qualité prix
3	éviter absolument	plutôt mécontente	globalement correcte	service client	très satisfait
4	mois après	voyez agent	pas années	très bonne	écoute besoins
5	non responsable	après ans	service client	bon service	rapport qualité prix
6	résilié contrat	bonus leurs	chaque année	très bon	aucun problème

FIGURE 6 – Bigrams et trigrams extraits par Naive Bayes par ordre d'importance, 0 étant la plus importante

Ces deux résultats nous donnent une idée des raisons de la satisfaction ou de l'insatisfaction des assurés. Le service client des assureurs est souvent mis en cause et notamment la lenteur de la réponse des assureurs par les clients mécontents. Étonnamment, les prix élevés ne sont pas les principales causes d'insatisfaction des personnes. Par ailleurs, le rapport qualité-prix est mis en avant par les assurés satisfaits.

Prédiction de la satisfaction d'une personne selon son commentaire

Dans le cadre de l'analyse d'une réputation d'un assureur, il est nécessaire d'appuyer l'analyse sur un grand nombre de documents. Toutefois, ces documents n'indiquent pas à priori s'ils reflètent l'avis de personnes satisfaites ou non satisfaites. On construit donc un algorithme qui, se basant sur la représentation vectorielle d'un document, détecte le niveau de satisfaction

de la personnes qui l'a écrit. Les résultats obtenus sont : la précision du classificateur est de 60% lorsqu'il s'agit de prédire exactement la classe de satisfaction (entre 1 et 5) et de 83% lorsque l'on prédit juste si un document exprime un sentiment plutôt positif ou au contraire un sentiment plutôt négatif. De plus, les erreurs fortes (prédiction de 5 au lieu de 1 par exemple) sont rares.

Création d'indicateurs sur la réputation d'un assureur

La réputation d'un assureur repose en partie sur la qualité des produits, le rapport qualité/prix et l'adéquation de l'assureur aux attentes des clients. La moyenne mensuelle de la satisfaction des individus permet de suivre si ces critères sont considérés satisfaisants par les assurés. On peut également collecter différents commentaires présents sur Internet (Twitter, Facebook,..) et estimer leur degré de satisfaction par le modèle précédemment créé puis les agréger à la moyenne. Cela donne une moyenne plus robuste.

Plus généralement, on peut chercher à suivre la "négativité" des commentaires postés par les personnes en s'intéressant à leur similarité à un ensemble de commentaires jugés très négatifs.

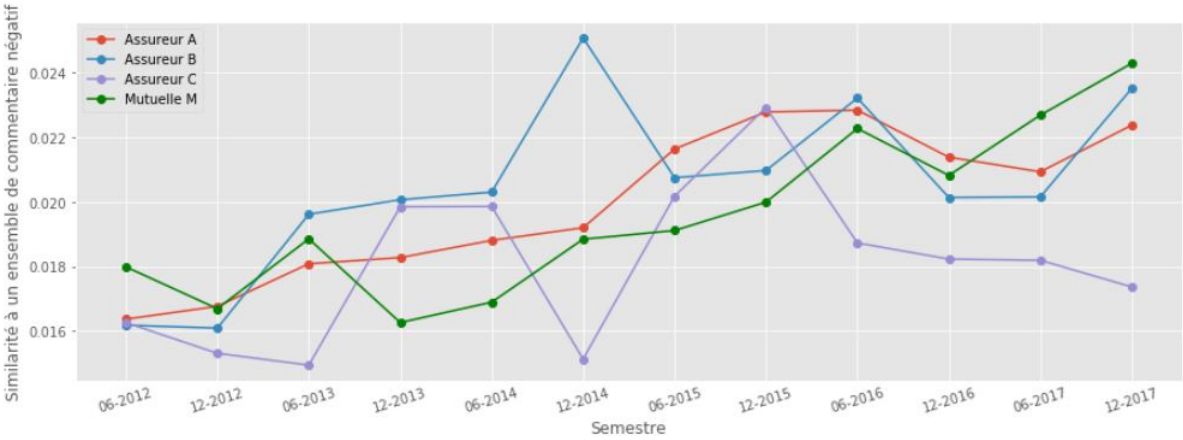


FIGURE 7 – Moyenne de la similarité des messages à un ensemble de messages négatifs par semestre, à partir de 2012

Ces graphes traduisent une augmentation continue du mécontentement des assurés envers leur compagnie d'assurance, sentiment évoqué dans cet article [2]. Le suivi de ces indicateurs peut permettre d'estimer les effets des politiques d'amélioration de la qualité de service des assureurs.

Conclusion

Dans ce mémoire, nous avons présenté et développé des algorithmes de reconnaissance d'écritures. Les réseaux de neurones peuvent être améliorés par une augmentation et une diversification de la base de données d'entraînement. Nous avons également montré quel pourrait être l'apport de ces algorithmes à des problématiques actuarielles.
Nous avons également développé des méthodes de compréhension automatique des demandes des assurés et créé des indicateurs de satisfaction des assurés envers leurs assureurs. Ces méthodes peuvent être utilisées dans le cadre plus général de l'analyse de la réputation d'un assureur.

2. `http://www.lefigaro.fr/conso/2016/12/27/20010-20161227ARTFIG00010-les-clients-sont-de-plus-en-plus`
`php`

Summary

Context

A Japanese life insurance company announced at the beginning of 2017 the replacement of 34 of its employees by artificial intelligence. These employees were dedicated to the study of clients' subscription files. This artificial intelligence has the ability to read documents relating to a person's subscription to a life insurance contract, to understand them and to decide quickly on pricing. Thus, tools and methods linked to artificial intelligence have to be appropriated by actuaries in order to control their risks.
Recognizing and understanding documents in order to automate decision-making processes can contribute to accelerate and improve the underwriting processes of insurance contracts. However, there are two types of documents : typescript (writen on computer)generally well processed by open source algorithms and handwritten script. Recognition of handwriting is still the object of scientific research, we present a method based on stat-of-the-art publications in data science and particularly in *Deep Learning*. Thanks to these methods, it is possible to extract textual information that text mining allows us to analyze. We explore some methods allowing the exploitation of unstructured data by challenging an example.

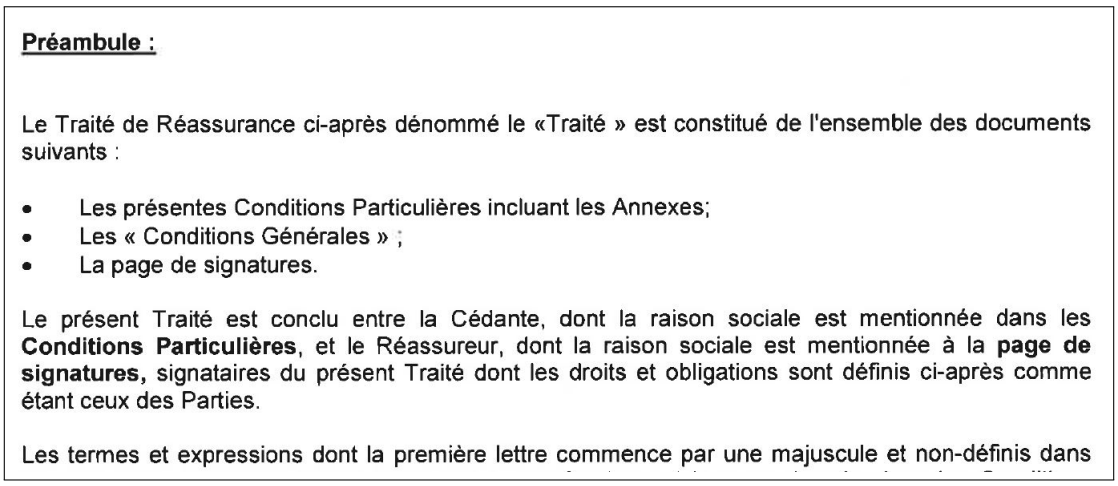
The new challenges of data science in character recognition and information retrieval

Data science in general and textit Deep Learning have revolutionized algorithms practicing optical character recognition (OCR). Some tasks such as letter or number recognition have improved significantly thanks to neural networks. Thanks to specifics neural structure (convolutional, recurrent, etc.), neural networks can recognize sequence of handwritted words. Text mining, sentiment analysis, or document classification then allows to make sense to extracted information from these documents.

Recognition of typed-writing

Insurers may own some type of scanned contract that is not directly exploitable. In order to automate their understanding, it is necessary to automatically recognize the typed text.
Characterized by its regularity, it is easily recognizable for some algorithms. These algorithms must include several steps :detecting the layout, segmenting text into connected components [3], splitting of the writing into letters and words and finally of text fonts.
Such algorithms are sometimes open source and are implemented in several packages.

3. fractions of image where the black pixels follow one another



Préambule :

Le Traité de Réassurance ci-après dénommé le «Traité » est constitué de l'ensemble des documents suivants :

- Les présentes Conditions Particulières incluant les Annexes;
- Les « Conditions Générales » ;
- La page de signatures.

Le présent Traité est conclu entre la Cédante, dont la raison sociale est mentionnée dans les **Conditions Particulières**, et le Réassureur, dont la raison sociale est mentionnée à la **page de signatures,** signataires du présent Traité dont les droits et obligations sont définis ci-après comme étant ceux des Parties.

Les termes et expressions dont la première lettre commence par une majuscule et non-définis dans

(a) Image

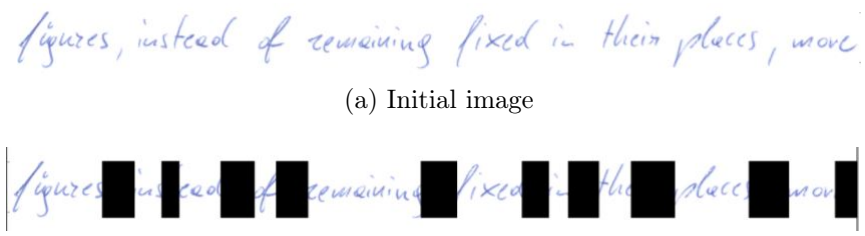
```
Préambule :

Le Traité de Reassurance ci-apres dénommé le «Traité » est constitué de l'ensemble des documents
suivants :

- Les présentes Conditions Particuliéres incluant les Annexes;
- Les « Conditions Générales » ,
- La page de signatures.

Le present Traité est conclu entre la Ce'dante, dont Ia raison sociale est mentionnée dans les
Conditions Particuliéres, et Ie Réassureur. dont Ia raison sociale est mentionnée a la page do
signatures, signataires du pre'sent Traité dont les droits et obligations sont definis ci—apres comme
étant ceux des Parties.

Les termes et expressions dont Ia premiere lettre commence par une majuscule et non-definis dans
```

(b) Recognized text

FIGURE 8 – Example of automatic analysis of scanned reinsurance contract

Handwriting detection and line and word segmentation

These methods that are performing well on typed script, are doing differently on handwritten text. However, if correspondences between insurers and their insured tend to be numerical, a lot of them are still handwritten for example auto insurance claims.
In some documents, only part of the document consists of handwriting. In order to detect and extract this part, an analysis of the thickness of the lines makes it possible to distinguish the handwriting from the typed writing and from other annotations. In general, the handwriting are thicker.
Once the handwritten text areas are recognized, Gaussian mixtures models can be used to segment them into lines and words. It is the first step of our analysis of documents. It is illustrated in the following figure :



(a) Initial image

(b) Recognized words

FIGURE 9 – Use of gaussian mixture model to segment in words

After the localization of images zones containing words, we can proceed to their recognition.

Recognition of Hidden Markov Modeling (HMM)

Each word image can be segmented into graphemes (small parts of images). Different of segmentation in letters, graphemes allows us to overcome the difficulty posed by crisscrossing characters.

Depending on their shape, graphemes are classified. Every word is then represented by a sequence of classes that we modeled by a hidden markov model (HMM). The HMM is a model that assumes that there is a hidden Markov chain that explain the observed sequence. Thus, the idea is to be able to modelize the chronological sequence of lineaments when writing a word. These models were the best in the early 2000s. However Deep Learning has changed the way computer vision is conceived and models more suited to our problem have emerged. Indeed, the HMM modeling doesn't take care of the gloabl context of a word. In fact, the prediction of the following graphemes is only based on the current graphemes.
Furthermore, the segmentation in graphemes is often delicate.Some Deep Learning approaches suggest new alternatives.

Recognition of neural network writing

Deep Learning or *deep learning* has become the norm in image processing today. Popularized by incredible results at the time in image recognition competitions (Imagnet), Deep Learning has gradually emerged as a major element in the processing of unstructured data (images, texts, sounds or also odor).

In the context of the recognition of handwriting, it is necessary to define the architecture of our deep learning model. In fact, several neural structures exists allowing to answer to our problematic. For example, the *feedforward network* is a network of simple neurons similar to a sequence of linear regressions. It can be trained on structured data (for example, the data of an insured person, his age, his sex, his guarantee, etc.) and respond to questions : is the insured risky ? what is its level of risk ? etc. The *convolutional neural networks* (CNN) are as far as they are concerned networks adapted to the analysis of images. They consist of layers of convolutions that have the ability to extract information on numerical matrices. Finally, *recurrent neural networks* have the possibility to take into account the sequential aspect of data. These are networks that have layers of neurons that communicate within themselves, thus taking into account the temporality of the data.

Based on litterature, we defined our own neural networb combining different neural layers. Thus, the general operation of the network can be summarized as follows : given the image, protruding points on the image will be detected thanks to convolutional layers and then the sequence of these points is modeled by recurrent neuron layers. Finally, a simple classification layer similar to a feedforward network but adapted to our problem (*CTC layer*) returns a probability matrix. From this matrix, we distinguish the recognized word. Following figure shows the complete structure of the network that is being trained :

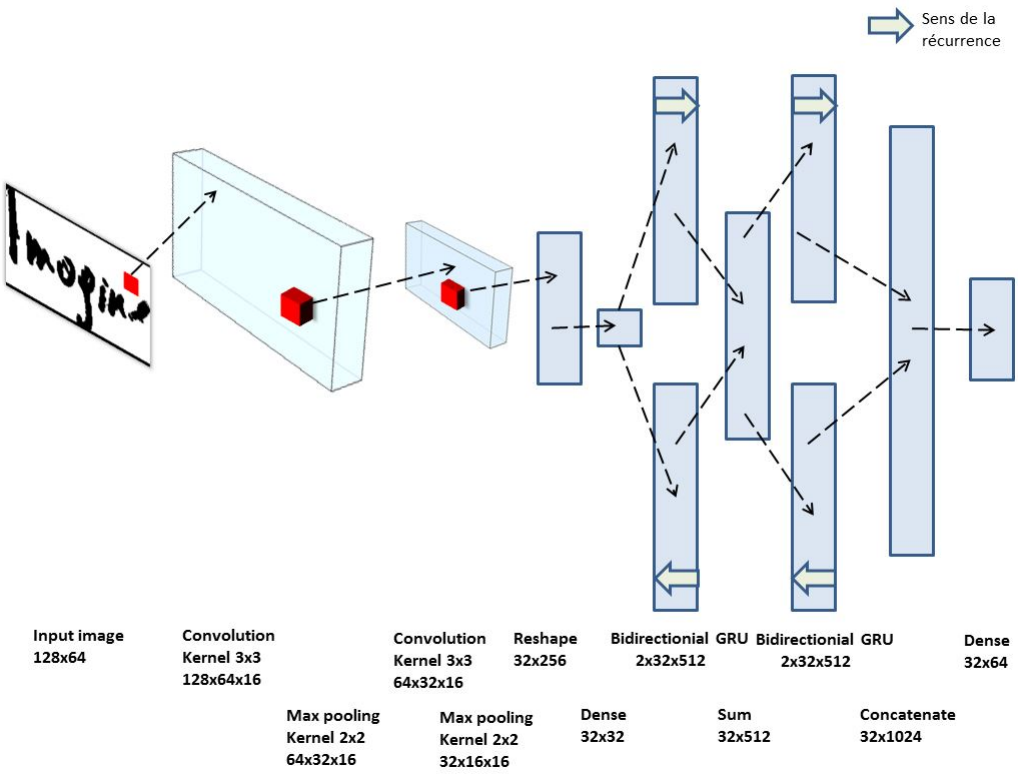


FIGURE 10 – Used neural network

The network includes 49 million parameters to optimize. This size led us to uses high capabilities of calculus using a large quantity of processor (GPU : Graphical Processor Units). This network is trained on different open source databases, consisting of 50,000 to 100,000 words images. Here is an example of results :

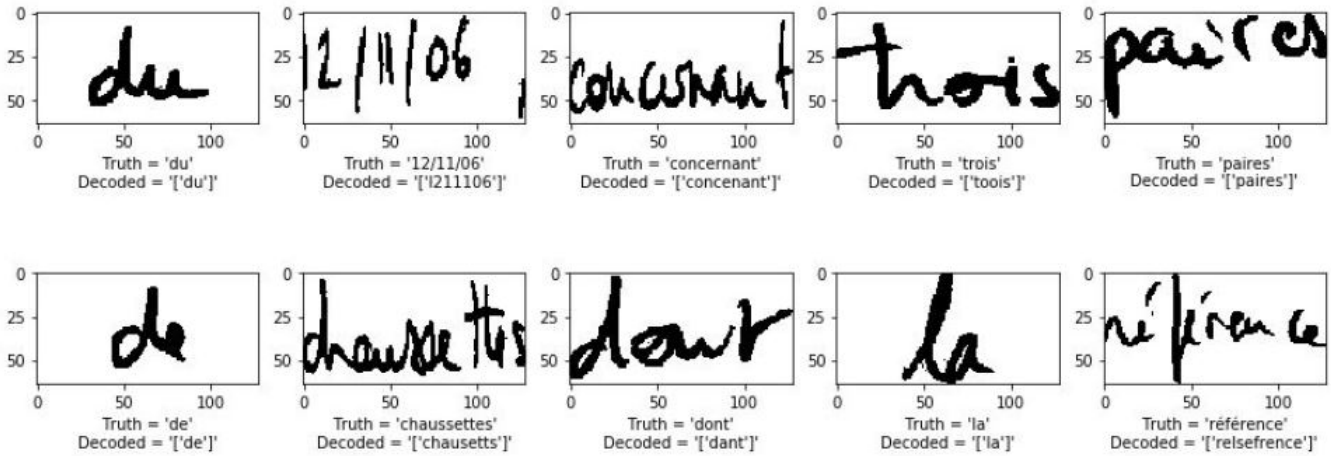


FIGURE 11 – Results of recognition of handwriting by neural networks

The results are generally satisfactory on a test data base similar to the training database. Nearly half of the words are fully recognized and a single well-calibrated spell-checker increases this score by correcting most mistakes. However, on words of different thickness or on images scanned with a low quality, the results degrade very quickly. This is due to the fact that the training database does not contain enough examples of various types.

Using the algorithm in different frames

Even if in current state of development, our algorithm can not be put in production, obtained results are hopeful. By taking care of different fonts of handwriting and by reinforcing the training of our algorithm, we are confident in the improving of its performances.
Beyond of the automatic analysis of auto claims, other frameworks of use of our algorithms can be envisaged. For example, the automatic reading of old death certificates for statistical studies on causes of death in the 1960s, the automatic study of birth certificates for exposure assessment in mortality table calculations, or yet, the reading of reinsurance contract scans for the pooling of guaranteed minimum rates.

Analysis of extracted information

Once text has been recognized, it should be automatically analyzed using text mining. We can not use directly the data from the first part, so we present text mining methods and make an application case on the analysis of the reputation of an insurer based on the comments posted on a specialized forum *opinion-assurances.fr*.

Text mining consists of a set of methods to form a numerical representation of words and to extract information from them. These methods are based either on a frequentist approach such as the textit tf-idf analysis, which tends to give a high weight to unsual words but frequent within the same document, or on a semantic approach and syntactic text. Two words will have similar vector representations if they have a close meaning. In the second case, it is networks of neurons that will create this representation. Moreover, word graphs can also provide an understanding of the relationships between words and a keyword extraction.

Extraction of keywords according to the satisfaction of the insured

The first analysis done on the insured's comments consists in extracting the keywords in order to understand the main points that the policyholders want to put forward. The extraction of the keywords can be done by an analysis on the word graph (analysis *k-core*) where a core is a subgraph where the nodes have a certain importance. Another methods consist in a Bayesian analysis of the representation tf-idf of each vector. The result using the first one, then looking for example the subgraph centered around the word "aucun" is this one :

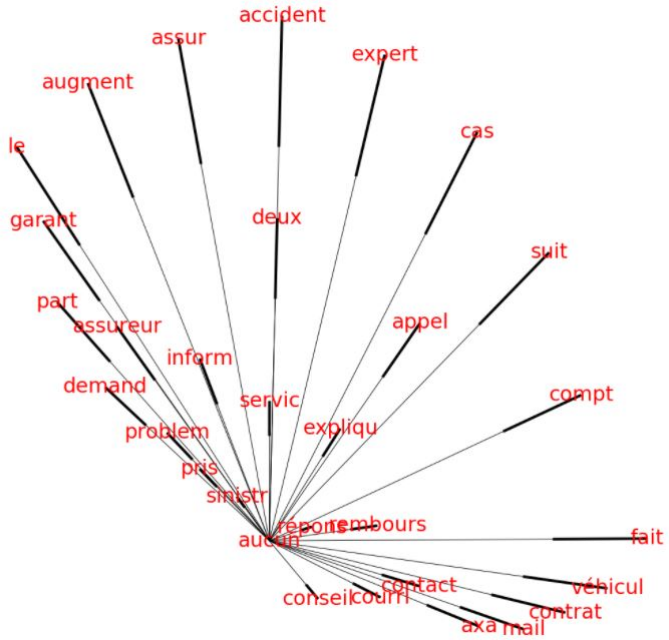


FIGURE 12 – Sub-graph centered around the word 'aucun'

Results using the *tf-idf* vectorial representation and attempting to identify the most important bigrams (couple of words) or trigrams give us this results :

Niveau de Satisfaction	**1**	**2**	**3**	**4**	**5**
Orde d importance					
0	service client	déconseille vivement	bonne mais	rapport qualité	satisfaite agence
1	fuir absolument	très déçue	ancien intermédiaire	qualité prix	très bon
2	depuis ans	manque professionnalisme	commentaire moins	rapport qualité prix	qualité prix
3	éviter absolument	plutôt mécontente	globalement correcte	service client	très satisfait
4	mois après	voyez agent	pas années	très bonne	écoute besoins
5	non responsable	après ans	service client	bon service	rapport qualité prix
6	résilié contrat	bonus leurs	chaque année	très bon	aucun problème

FIGURE 13 – Extracted bigrams and trigrams by Naive Bayes in order of importance, 0 being the most important

These two results give us an idea of the reasons for the satisfaction or dissatisfaction of the insured. The customer service of insurers is often questioned and in particular the slowness of the response of the insurers by dissatisfied customers. Surprisingly, high prices are not the main causes of people dissatisfaction. In addition, the quality-price ratio is highlighted by the satisfied insured.

Prediction of satisfaction of a person according to his / her comment

In analyzing an insurer's reputation, it is necessary to support the analysis on a large number of documents. However, these documents do not indicate a priori whether they reflect the opinion of satisfied or dissatisfied persons. An algorithm is thus constructed which, based on the vector representation of a document, detects the level of satisfaction of the person who wrote it. The

results obtained are : the accuracy of the classifier is 60 % when it is a matter of accurately predicting the satisfaction class (between 1 and 5) and 83 % when one predicts just whether a document expresses a feeling rather positive or rather a rather negative feeling. Futhermore, the strong errors (prediction of 5 instead of 1 for example) are rare.

Creating indicators on an insurer's reputation

The reputation of a company depends in part on the quality of the products, the quality / price ratio and the suitability of the company to the expectations of the customers. The monthly average of the satisfaction of the individuals makes it possible to follow if these criteria are considered satisfactory by the individuals. We can also collect different comments on the Internet (Twitter, Facebook, ...) and estimate their level of satisfaction by the previously created model and then aggregate them to the average. This gives a more robust average. More generally, one can seek to follow the "negativity" of the comments posted by the people by looking at their similarity to a set of comments deemed very negative.

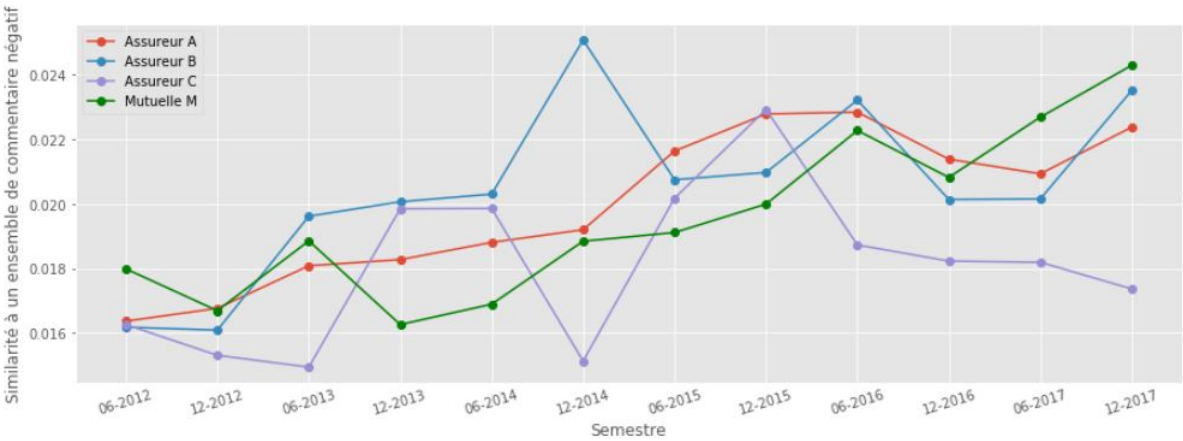


FIGURE 14 – Mean of similarity between a commentary with a set of negative commentary by semester, since 2012

We see for all insurers and health mutuals a continuous increase of the dissatisfaction of the people, reflecting the effect explained in this article [4]. Monitoring these indicators can help to estimate the impact of insurers' quality of service improvement policies.

Conclusion

In this paper, we have introduced and developed writing recognition algorithms. Neural networks can be improved by increasing and diversifying the training database. We have also shown how these algorithms can contribute to actuarial problems.
We have also developed methods for automatically understanding insured claims and creating indicators of satisfaction of policyholders with their insurers. These methods can be used in the more general framework of an insurer's reputation analysis.

4. http://www.lefigaro.fr/conso/2016/12/27/20010-20161227ARTFIG00010-les-clients-sont-de-plus-en-plus
php

Table des matières

Table des figures

Liste des tableaux

1 Introduction

5 Janvier 2017, une compagnie d'assurance vie japonaise, la "Fukoku Mutual Life Insurance" communique sa décision de remplacer trente-quatre de ces employés ayant pour mission l'étude des dossiers médicaux de potentiels assurés par une intelligence artificielle développée par IBM [6] [7]. Cette dernière serait capable d'étudier les dossiers médicaux et de calculer le paiements des primes de manière autonome, avec toutefois un opérateur humain pour valider les montants. Cette annonce est l'une des premières de ce qui semble être une tendance qui s'amplifie chez les assureurs. L'incursion de la data science dans le métier de l'actuariat a en effet impacté la façon de penser l'assurance [8] [9] [10]. Ainsi, l'actuaire d'aujourd'hui est amené à s'approprier les outils et méthodes de l'intelligence artificielle afin d'en maîtriser les risques, mais surtout d'être en adéquation avec les nouveaux besoins des assurés de plus en plus connectés.

"L'information est le pétrole du 21ème siècle et l'analyse de données en est la machine à combustion", c'est par ces mots que Peter Sondergaard, directeur de recherche chez Gartner, définit la place du *Big Data* et de la data science dans l'ère moderne. Parmi les récentes avancées de la data science, les différentes méthodes pour traiter les données non-structurées [11] telles que les données textuelles ou les images ont considérablement progressé. Les résultats obtenus dans la reconnaissance ou la détection d'objets, de sentiments sont désormais quasiment au niveau des compétences humaines. [12]

Précisément, les assureurs ont aujourd'hui des masses de données non-structurées difficilement exploitables par un ordinateur ou un algorithme sans retraitement. C'est le cas, par exemple, de contrats de réassurance, de conditions générales de contrat de prévoyance, de rapports d'accidents domestiques ou industriels, de constats automobiles, de dossiers médicaux qui sont des documents que les assureurs ou réassureurs cumulent sous forme de documents scannés. L'exploitation manuelle de ces données est humainement coûteuse et leur lecture par un opérateur pour l'extraction d'informations et pour leur classification demande un temps important. Ce temps impacte la gestion opérationnelle des compagnies d'assurance et l'efficacité de leurs prises de décisions. De même, les fraudes sont plus facilement détectables avec une analyse exhaustive de ces données.

Dans ce mémoire, nous nous focaliserons particulièrement sur la reconnaissance des documents contenants du texte. Ces documents sont ou bien des documents tapuscrits, c'est à dire des documents édités par ordinateur, ou bien des documents manuscrits, écrits à la main par des assurés en général. Pour exploiter automatiquement ces données, des algorithmes pratiquant la reconnaissance optique de caractères (OCR pour *Optical Character Recognition*) ainsi que des méthodes utilisées en data science telles que le *Deep Learning* vont être présentés et utilisés. A noter qu'il est indispensable de considérer des méthodes différentes pour l'écriture tapuscrite et l'écriture manuscrite. Cette dernière nécessite l'utilisation d'algorithmes robustes à la typologie de l'écriture (parfois difficile pour un humain à analyser) et à la qualité de l'image.

6. http://www.numerama.com/tech/221747-une-i-a-remplace-34-employes-dune-assurance-au-japon.html
7. http://bfmbusiness.bfmtv.com/entreprise/assurance-l-intelligence-artificielle-va-t-elle-remplacer-html
8. http://www.argusdelassurance.com/acteurs/le-nouveau-big-bang-de-l-actuariat.67806
9. http://www.thesuperposteapourvoir.com/2017/02/le-big-data-et-lactuariat-dans-lassurance/
10. http://www.atlas-mag.net/article/intelligence-artificielle-et-assurance
11. Données ne pouvant être mises dans un tableau suivant une structure
12. Voir par exemple, les résultats sur la base de données MNIST http://yann.lecun.com/exdb/mnist/

Dans un second temps, il convient d'analyser l'information textuelle extraite pour en extraire un signal utilisable par l'assureur. La mise en avant de mots-clés, la détection de sentiments prédominants et plus généralement le *text mining* sont des moyens techniques qui permettent l'extraction d'informations à partir de documents. Nous mettrons en avant leur utilisation dans le cadre de l'analyse de la réputation d'un assureur.

Des millions d'internautes postent quotidiennement des commentaires et interagissent avec les entreprises par le biais de forums spécialisés ou encore sur les réseaux sociaux. Ces échanges, parfois facilement consultables et téléchargeables, peuvent être exploitées pour caractériser la réputation d'une entreprise ou détecter un sentiment prédominant chez les consommateurs. Dans le cadre de l'analyse d'un risque de réputation pour les assureurs, nous allons nous intéresser à des données provenant d'un forum spécialisé dans le secteur assurantiel [13]. Cette évaluation peut constituer la base de la prise en compte du risque de réputation pour les assureurs.

L'idée de ce mémoire est donc de présenter, de proposer et d'implémenter des algorithmes automatisant la reconnaissance de documents scannés et l'exploitation de textes provenant de données assurantielles en s'appuyant sur la data science.

Dans la première partie, nous allons nous intéresser aux différentes méthodes pour reconnaître l'écriture dactylographiée et l'écriture manuscrite. Nous présenterons les méthodes les plus pertinentes qui ont été à l'origine de résultats prometteurs et les méthodes les plus récentes dont l'utilisation du *Deep Learning*. Nous lèverons particulièrement le voile d'incompréhension qui recouvre le Deep Learning actuellement en explicitant les modèles associés. Nous mènerons également des expériences sur des bases de données publiques et semi-publiques afin d'évaluer la qualité des modèles définis.

Dans la seconde partie, nous mènerons des analyses textuelles sur des données directement récupérées sur des forums spécialisés en assurance. Nous présenterons les méthodes associées au text mining de manière générale et nous verrons qu'il est possible de les utiliser à des fins d'analyse d'une réputation.

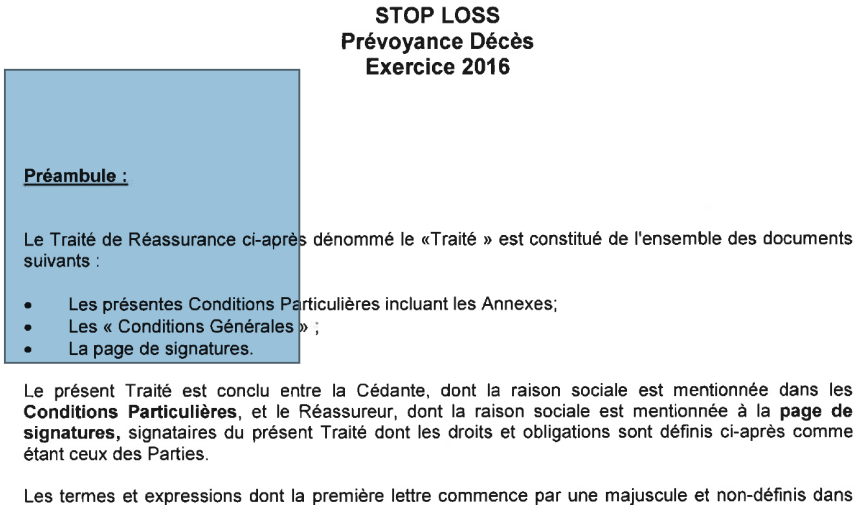
13. https ://www.opinion-assurances.fr/

2 L'extraction d'information à partir de données non structurées

2.1 Contexte et état de l'art en assurance

Dans le cadre de leurs relations avec leurs clients, les assureurs peuvent conserver une trace écrite de l'ensemble des contrats de souscription en cours et passés. De même que les documents relatifs à des déclarations de sinistres, constats amiables ou encore rapports d'experts, ces documents peuvent être stockés par l'assureur sous la forme de documents scannés. Par ailleurs, en assurance vie, les rapports d'analyse, les compte-rendus de visites médicales, les certificats et autres questionnaires sont souvent des documents contenant de l'écriture manuscrite que souvent seul un opérateur dédié analyse.

Il n'est effectivement pas rare que la plupart de ces documents soient scannés par les assurés et envoyés directement aux compagnies d'assurance [14]. Ces documents numérisés sont donc non reconnus comme possédant du texte par l'ordinateur, comme le montre la figure 15 et ne peuvent faire directement, par exemple, l'objet d'une recherche par mot-clés.



STOP LOSS
Prévoyance Décès
Exercice 2016

Préambule :

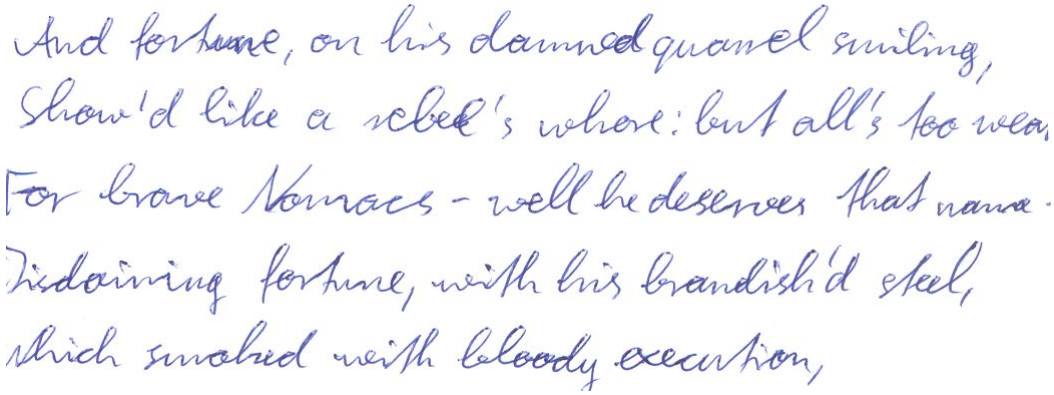
Le Traité de Réassurance ci-après dénommé le «Traité » est constitué de l'ensemble des documents suivants :

- Les présentes Conditions Particulières incluant les Annexes;
- Les « Conditions Générales » ;
- La page de signatures.

Le présent Traité est conclu entre la Cédante, dont la raison sociale est mentionnée dans les **Conditions Particulières**, et le Réassureur, dont la raison sociale est mentionnée à la **page de signatures,** signataires du présent Traité dont les droits et obligations sont définis ci-après comme étant ceux des Parties.

Les termes et expressions dont la première lettre commence par une majuscule et non-définis dans

(a) Texte tapuscrit



And fortune, on his damned quarrel smiling,
Show'd like a rebel's whore: but all's too weak,
For brave Macbeth – well he deserves that name –
Disdaining fortune, with his brandish'd steel,
which smoked with bloody execution,

(b) Texte manuscrit

FIGURE 15 – Exemple de document scanné où la sélection de texte est impossible

De manière générale, afin d'exploiter ces données, les assureurs ont recours à des opérateurs qui lisent ces documents les uns après les autres. Le temps pris par ces tâches, peut se répercuter

14. au format JPG, JPEG, PNG ou encore PDF

sur différents aspects de la gestion des dossiers par l'assureur [15]. Dans le cadre de l'assurance non-vie par exemple, les assureurs doivent répondre aux sinistres qui leurs sont déclarés par leurs assurés. Les assureurs ont donc mis en place des procédures afin de prendre les bonnes décisions (de remboursement, d'envoi d'un expert par exemple) concernant chaque sinistre le plus rapidement possible. Ces processus ont été optimisés dans le but d'améliorer le service client par différents moyens mais l'intervention humaine reste nécessaire. Par ailleurs, les décisions de remboursement sont parfois trop rapides, engendrant des erreurs et donc des pertes pour l'assureur. L'intelligence artificielle peut donc répondre à ce besoin de vitesse et de précision en automatisant certaines tâches (par exemple, la lecture et la classification des dossier).

Pour les contrats d'assurance vie, les délais de souscription sont généralement importants, de l'ordre de 30 jours en moyenne. Étant déjà particulièrement contraignants pour la quantité de documents à réunir et à envoyer, le processus de souscription peut décourager les potentiels assurés, ce qui engendre une perte pour l'assureur. Par ailleurs, afin d'accélérer ce processus, parfois les justificatifs demandés par les assureurs ne sont pas lus , entrouvrant ainsi une porte à la fraude. C'est pourquoi la compagnie d'assurance vie *Prudential* a proposé un data challenge sur Kaggle [16]. Prudential fait le constat que seulement 40% des ménages américains ont souscrit à un contrat d'assurance vie et met en cause les difficultés du processus de souscription. Le data challenge a pour but la création d'un modèle qui permet d'estimer de manière automatique la catégorie de risque dans laquelle se trouve une personne en se basant uniquement sur quelques données. Ce challenge a aussi pour objectif de permettre à Prudential de comprendre la puissance de prédiction de ces données.

L'intelligence artificielle a donc un attrait particulier pour les assureurs si elle permet d'accélérer les processus de décisions en même temps que d'améliorer la qualité des décisions.

C'est la motivation de la Fukoku Mutual Life Insurance qui a annoncé vouloir utiliser une intelligence artificielle pour exploiter les documents des assurés. Cette technologie pourrait augmenter de 30% l'actuelle capacité du service qu'elle souhaite remplacer, tout en étant exhaustive sur la lecture de documents fournis à l'assureur.

Par ailleurs, nos recherches bibliographiques laissent penser que les assureurs et le monde de la recherche actuarielle restent en retrait, pour l'instant, sur ce type de problématique. Il semble cependant nécessaire que ce soit des actuaires au fait des techniques de la data science qui développent ces outils de reconnaissance pour avoir une capacité d'adaptation aux attentes des assureurs et une réelle vigilance quant aux risques engendrés par ce type de technologies. C'est une raison qui motive alors la problématique abordée dans ce mémoire.

Plus généralement, l'assureur, afin d'assurer sa pérennité, doit suivre un ensemble de règles strictes quant à la gestion des risques qu'il assure. Ainsi son risque de faillite doit être mesuré et contrôlé de manière précise. Le calcul de ce risque doit s'établir en tenant compte de l'ensemble des risques pris par l'assureur dont le *risque opérationnel*.

Dans leur article [7], Dalla Valle et al. donnent la définition qui suit du risque opérationnel : "Une définition plus précise des risques opérationnels comprend les pertes directes ou indirectes engendrées par l'insuffisance ou le mauvais fonctionnement des procédures, des ressources humaines et des systèmes internes, ou par des événements extérieurs. Au fond, il s'agit

15. `http://www.mondaq.com/unitedstates/x/366524/Insurance/Why+It+Takes+So+Long+For+Insurance+Carriers+To+Respond+To+A+Claim+And+What+You+Can+Do+About+It`
16. `https://www.kaggle.com/c/prudential-life-insurance-assessment`

dans tous les cas de pertes imputables à des erreurs humaines, à des problèmes techniques ou de procédure, ou à d'autres causes non reliées au comportement des opérateurs des marchés financiers ou à des événements survenus sur les marchés".

L'intelligence artificielle a pour vocation de réduire le risque opérationnel et plus précisément celui qui est liée aux erreurs de saisie [17] qui entraînent des décisions erronés et des erreurs de jugement. La conviction de l'assureur japonais Fukoku Mutual Life Insurance est qu'il va réduire son risque opérationnel en remplaçant ces opérateurs de saisie par des algorihtmes.

2.2 État de l'art en Data-Science

La reconnaissance d'écriture manuscrite a longtemps été un enjeu pour la science. Par sa complexité, la recherche dans cette discipline a peu évolué jusqu'aux années 2000. Ce n'est que grâce à l'émergence des réseaux neuronaux [18] que cette discipline fait l'objet de nouvelles recherches ces dernières années. En effet, fasciné par l'idée de répliquer le comportement du cerveau humain, les chercheurs ont réussi à analyser de nombreux signaux grâce aux réseaux de neurones profonds. La Reconnaissance Optique de Caractères (ou *Optical Character Recognition* - OCR en anglais) n'y échappe pas et la compréhension des mécanismes humains inspire la création de modèles : au commencement de leur vie, la plupart des individus développent une capacité à déchiffrer et à comprendre une écriture manuscrite malgré les différences de calligraphies des auteurs. C'est cette capacité que la science a voulu imiter par l'intermédiaire de modélisation toujours plus pointues et des techniques de plus en plus affinées. L'idée qui a motivé un nombre important de scientifiques, était donc de faire acquérir à l'ordinateur des capacités humaines. Cette idée fascinante par bien des aspects émerge dès le début des années 50 et se poursuit dans les années 60, 70 avec un intérêt réel des entreprises, banques, hôpitaux et administrations qui y voient des possibilités importantes. De plus, l'utilisation de la machine à écrire puis de l'ordinateur dans la création et le traitement de texte, implique que la plupart des textes deviennent tapuscrits (et non plus manuscrits). Ces deux formes d'écritures se traitent différemment, l'une étant plus régulière avec des espaces entre les caractères, l'autre étant plus aléatoire et donc plus compliquée à déchiffrer.

L'avancée des mathématiques durant des vingt dernières années d'une part et l'accroissement rapide de la puissance des ordinateurs va permettre l'émergence de modèles mathématiques complexes avec des techniques d'optimisation avancées. C'est dans ce foisonnement intellectuel d'études, de recherches scientifiques et de projets de recherche innovants, que les mathématiques vont être appliquées à des problèmes de plus en plus variés dont l'OCR. Les problèmes séquentiels en général comme le séquencage de l'ADN, la reconnaissance audio, ou encore l'OCR vont être largement traités. Les performances vont être continuellement améliorées. Plus récemment, l'explosion de la recherche dans la Data-Science et l'émergence du Deep Learning ont permis de franchir un nouveau palier. Ainsi aujourd'hui des résultats pouvant atteindre ceux d'opérateurs humains sont obtenus.

De ce fait, historiquement, les premiers modèles de reconnaissance d'écriture existants ([9], [29], [30]) tente d'utiliser les formes géométriques des lettres pour les reconnaître. Ces modèles sont rapidement limités par la multiplicité des styles d'écriture, propres à chaque individu.

Des algorithmes plus récents sont ensuite basés sur une modélisation statistique de l'écriture.

17. `http://obsmetiers.rcp-pro.fr/fileadmin/observatoire_metiers/documents/etudes/Fiche_de_synthese.pdf`
18. Rendue possible grâce à l'utilisation des nouvelles technologies informatiques : CPU, GPU, TPU.

Le *modèle de markov caché* (Hidden Markov Model) [17] est ainsi l'un des premiers à prendre en compte le fait que l'écriture n'est pas seulement un assemblage de lettres quelconques mais une séquence de lettres qui se suivent intelligemment. Le modèle de Markov caché est un modèle statistique dans lequel le système modélisé est supposé être un processus markovien de paramètres inconnus. Les HMM ont été décrits pour la première fois dans une série de publication de statistiques par Leonard E. Baum et d'autres auteurs après 1965. Plus récemment, le Deep Learning a obtenu des résultats impressionnants dans la reconnaissance. Des réseaux de neurones dont l'architecture prend en compte l'aspect séquentiel des mots sont maintenant les techniques utilisées pour l'OCR.

Ainsi, la plupart des systèmes la reconnaissance d'écritures tapuscrite ou manuscrite capturé par un scanner ou par tout autres systèmes optique, l'image digitale passe à travers les étapes suivantes d'un système de reconnaissance par ordinateur :

1. L'étape de *preprocessing* permettant d'améliorer la qualité de l'image et de localiser les parties intéressantes de l'image.
2. L'extraction de *features* qui contiennent les caractéristiques qui permettent de distinguer les caractères.
3. La classification qui va utiliser les features pour identifier les mots et caractères.

Dans la suite, nous allons présenter et implémenter les modèles utilisées en reconnaissance d'écriture les plus courants et les plus modernes. Nous ferons une différence entre l'écriture tapuscrite où nous présenterons différentes méthodes déjà existantes et très performantes et l'écriture manuscrite où nous proposons et entraînons un réseau de neurones. Nous verrons que l'étape 2 et l'étape 3 précédemment citées peuvent être regroupées désormais grâce au Deep Learning. Nous expérimenterons nos modèles pour l'écriture manuscrite sur des bases de données existantes.

2.3 Les enjeux méthodologiques de l'extraction et de la reconnaissance de l'écriture tapuscrite

La reconnaissance de document tapuscrit est plus simple que les documents manuscrits du fait que l'écriture est davantage régulière et que les caractères sont facilement séparables. Il n'y a ni chevauchement de lettres ni caractères difformes qui rendrait la reconnaissance plus complexe.

Le problème de la reconnaissance d'écriture tapuscrite a été résolu il y a quelques années. Certaines entreprises spécialisé dans le matériel informatiques d'impression et de scan [19] ont utilisé leurs laboratoires de recherches à cette fin. Ainsi en 1984, Ray Smith [20], afin de développer une technologie pouvant produire des résultats corrects. Sa technologie sera ensuite reprise par Google en 2005 et développée pour donner un module du nom de Tesseract. On peut également citer la technologie développée par la compagnie russe *Cognitive Technologies* appelée *CuneiForm* [21]. Ce sont toutes des technologies très performantes.

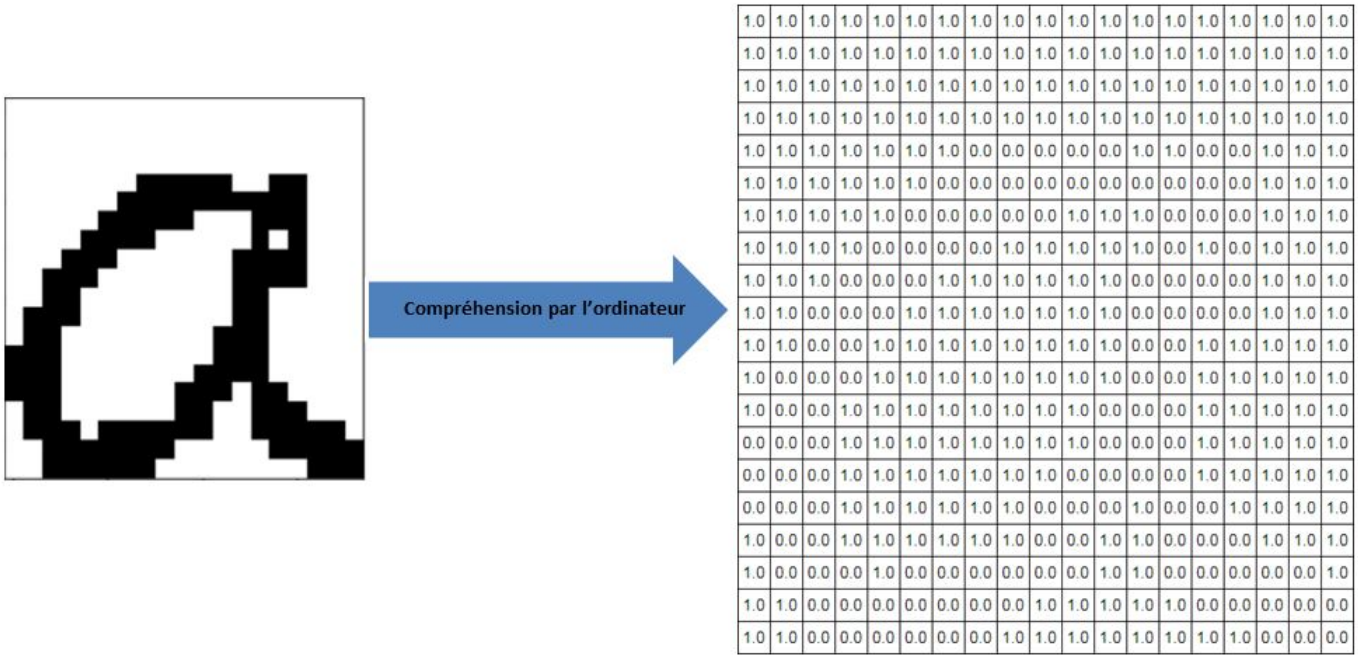
Nous avons attaché une importance particulière à la compréhension des techniques utilisées par ces modules. Ainsi, nous proposons de décortiquer point par point les phases d'analyse automatique d'une écriture numérique. Nous décrirons de manière générale les étapes nécessaires à la reconnaissance d'écriture tapuscrite et nous en utiliserons certains pour reconnaître quelques

19. Hewlett Packard, Epson,...
20. Dans les laboratoires d'Hewlett Packard,voir lien
21. `https://en.wikipedia.org/wiki/CuneiForm_(software)`

documents. Nous mettons également en place une application qui transforme directement un fichier PDF de plusieurs pages composées de documents scannées en un fichier PDF composés de documents scannées où la recherche de texte est possible.

2.3.1 Définition mathématique d'une image

Cette partie du mémoire est consacrée à l'extraction d'informations à partir d'images. Définissons ce qu'est une image et sous quelle forme un ordinateur la représente. Une image est [22]la "reproduction d'un objet matériel donnée par un système optique et, en particulier, par une surface plane réfléchissante ou un miroir". Cette reproduction peut être stockée de différentes manières, soit sur sur du papier, soit numériquement. Plus concrètement, une image peut être perçue comme une matrice. Chaque coordonnée correspond alors à un pixel [23].
La figure 16 illustre la représentation numérique d'une image en noir et blanc.



1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0	1.0	1.0	1.0
1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	0.0	1.0	0.0	1.0	1.0	1.0
1.0	1.0	1.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0
1.0	1.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0
1.0	1.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0
1.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0
1.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0
0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0
0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0
0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	1.0	1.0	1.0
1.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	1.0	1.0	1.0
1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0
1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0
1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0

FIGURE 16 – Lecture d'une image par un ordinateur

Pour une image en niveau de gris, les valeurs dans la matrice sont comprises entre 0 et 255. Chaque nombre entier entre 0 et 255 représente un niveau de gris. Pour une image de couleur, dans le cas d'une représentation d'image en convention Rouge Vert Bleu par exemple, chaque coefficient de la matrice est constitué de trois chiffres, le premier indiquant le niveau de rouge, le second indiquant le niveau de bleu et le troisième indiquant le niveau de vert dans l'image.

Nous retiendrons cette représentation matricielle comme la donnée en entrée pour chacun de nos programmes de reconnaissance, pour l'écriture manuscrite comme pour l'écriture tapuscrite.

---

[22] définition du Larousse

[23] i.e le plus petit élément constitutif d'une image produite ou traitée électroniquement, définie par sa couleur et sa luminosité

2.3.2 Constitution d'un programme d'OCR

La première nécessité d'une technologie capable de faire de l'OCR est la possibilité qu'elle distingue la zone de texte, des zones sans texte et de lire l'écriture blanche sur fond gris comme l'écriture grise sur fond blanc. Certaines technologies sont architecturées de manière à d'abord extraire les grandes lignes des images noirs-et-blancs quelle que soit la couleur de fond de l'image puis ensuite à extraire des caractéristiques à partir de ces grandes lignes, ce qui assure une certaine invariabilité vis-à-vis de la couleur de la page.

Pour avoir une reconnaissance de qualité sur des documents de types différents, le système de reconnaissance doit posséder une architecture avec des blocs ayant chacun une ou plusieurs fonctionnalités (voir Fig.17) et ayant la possibilité de s'adapter automatiquement au document fourni comme entrée.

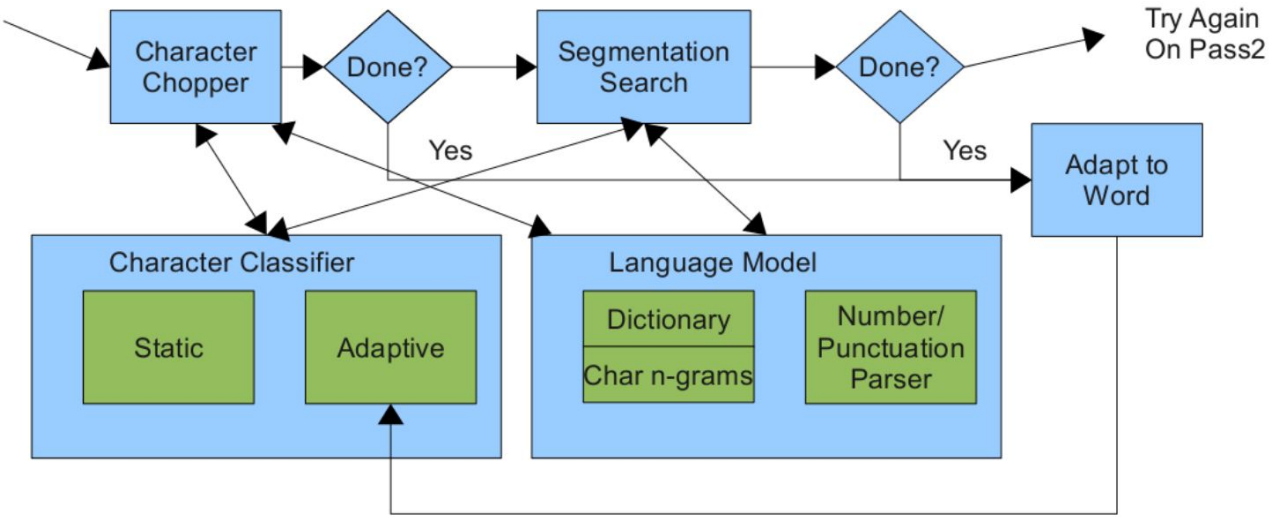


FIGURE 17 – Exemple de système de reconnaissance de mots, Ray Smith, DAS 2014

2.3.3 Pipeline d'un programme d'OCR

L'image fournie au programme doit tout d'abord être analysée pour détecter la mise en page du document. Cette étape est essentielle dans la lecture de documents de tous types, journal, livre, lettre, rapport, etc. où la mise en page peut être complexe. L'analyse qui suit va alors se baser sur les seules zones de texte :

1. La première étape est une *analyse en composante connectée* qui permet de retrouver les différentes pixels noirs d'une image qui se suivent [24] de l'image.
2. A partir des composantes connectées, une recherche de lignes d'écriture est effectuée.
3. Ces lignes vont être ensuite analysées et découpées en mots selon l'espace entre les caractères.
4. Ensuite une reconnaissance des mots en deux temps se produit. L'algorithme parcourt une première fois le document et essaie de reconnaître chaque mot. Chaque mot reconnu avec une taux de satisfaction élevé [25] est conservé dans une base de donnée. Cette base est ensuite utilisée pour reconnaître avec plus de précisons les mots suivants.

24. définition d'une composante connectée
25. i.e avec une probabilité élevée

5. Afin de ne pas avoir une qualité de reconnaissance plus faible au début du document qu'à la fin, puisque la base de données était alors encore de petite taille, l'algorithme va parcourir le document une seconde fois. L'algorithme est ainsi dit adaptatif.
6. Finalement une étape de confirmation permet de résoudre les mots dont le découpage est incertain et vérifie la taille de la police.

2.3.4 Analyse en composantes connectées

Cette analyse permet d'extraire des parts d'image souvent proches des lettres. Il existe différents algorithmes d'analyse en composantes connectées qui vont étudier l'ensemble des pixels et détecter un ensemble de pixels connectés. La définition de deux pixels connectés est donné par la 4 (ou 8) -connexité. Deux pixels sont connectés s'ils ont la même couleur et si les deux pixels se suivent horizontalement et verticalement pour la 4-connexité ainsi que diagonalement pour la 8-connexité [4]

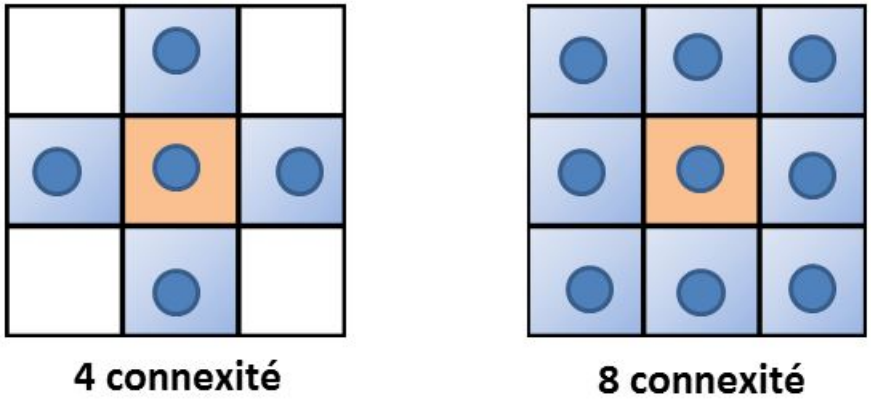


FIGURE 18 – Connexité

Des algorithmes simples qui parcourent l'ensemble de l'image et qui recherche des ensembles de composantes connectées obtiennent de bons résultats. Toutefois sur des images de grandes tailles, ces algorithmes ne sont plus performants. D'autre approches sont alors utilisés, comme celle de la théorie des graphes. A partir d'une image, on peut construire un graphe où les noeuds sont les pixels de couleur et les ponts sont définis si une certaine distance entre les noeuds n'est pas dépassé. A partir de ces graphes, un clustering des noeuds est réalisable. Voici les résultats obtenus sur une image de couleur :

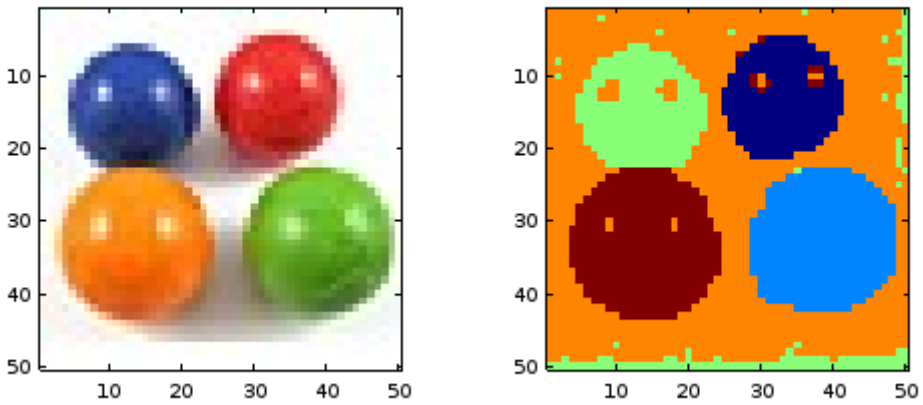


FIGURE 19 – Résultat d'une analyse en composantes connectées en utilisant la théorie des graphes

2.3.5 Recherche de lignes d'écritures

Après avoir obtenues les composantes connectées (petites parties de l'image), on cherche à distinguer les lignes d'écritures. Cette étape débute donc par un filtrage des composantes connectées détectées. Toutes les composantes ayant une taille plus petite qu'une fraction de la médiane de la taille des composantes, sont retirées (uniquement pour cette étape). Par cette méthode, on s'attend ainsi à éliminer les composantes connectées dues à de la ponctuation et au bruit.

Les composantes filtrées vont ensuite être toutes assignées à une ligne selon leurs positions dans l'image. Une fois toutes les composantes assignées à une ligne, une régression des moindres carrés médians donne une idée exacte de la base de la ligne d'écriture (*baseline*), les composantes sont alors réassignées de manière certaine selon ces baselines. Un affinement de ces lignes est ensuite effectué pour envisager le cas de lignes courbes, ce qui peut arriver sur des documents scannées. Finalement une fusion des composantes qui s'entrecroisent horizontalement (dans l'orientation de la ligne retrouvée) au moins de moitié permet de recomposer des lettres coupées.

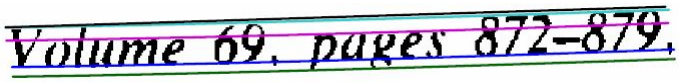


FIGURE 20 – Exemple d'une ligne de base incurvée (en bleu), [31]

2.3.6 Recherche de mots et de lettres

Une fois les lignes d'écritures repérées, l'algorithme va chercher à distinguer les mots et les lettres selon chaque ligne d'écritures. Il existe différentes façons de réaliser cette étape. Dans le cadre de l'écriture manuscrite 3.4, une autre méthode est présentée. L'algorithme [26] va tester dans un premier temps les lignes d'écritures pour déterminer s'il existe un espacement régulier. Si cet espacement existe, alors le découpage en lettres est simple. S'il n'existe pas d'espacement régulier, ou si celui-ci n'est pas évident, Le module mesure alors l'ensemble des distances horizontales entre pixels de couleur noirs entre la baseline (en bleu sur la figure 20) et la ligne médian (en rouge sur la figure 20). Il va alors définir deux seuils. L'un au dessous duquel, l'espacement est dû à l'espace entre deux caractères, et l'autre au dessus duquel l'espacement est

26. c'est le cas des modules open source, Tesseract ou encore CuneiForm

du à un espace entre deux mots. Entre ces seuils, l'espacement est considéré comme incertain et sera évalué à la dernière étape.

2.3.7 Reconnaissance des mots

L'algorithme possède maintenant en mémoire les images de chaque lettre d'un document écrit à l'ordinateur. La reconnaissance des mots peut alors se faire lettre à lettre. Indépendamment du document et en amont de l'utilisation de l'algorithme, on constitue une base de données des points saillants de l'image de lettres de tout type. Ces points saillants sont des morceaux de l'approximation polygonale de l'image de ces lettres. Elles constituent le *prototype* de la lettre [27]. Puis on extrait de manière automatique des features de la lettre à reconnaître. Ces features ne sont pas les mêmes que celles précédemment ! L'idée géniale ici est que l'on s'intéresse à une distance entre les features et non pas à les faire matcher exactement, ce qui permet d'avoir un extracteur de features sur les lettres inconnues régulier et consistant.

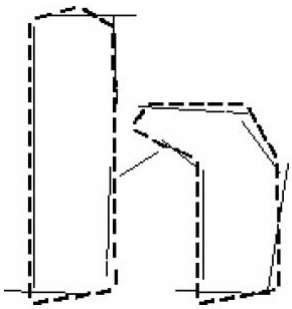


FIGURE 21 – Exemple de features et d'un prototype, [31]

Dans la figure 21, les petits trais en gras sont les features extraits sur la lettre à reconnaître et les traits longs et fins sont les features issues du prototype.

La limite à cette approche est le nombre de calculs à effectuer pour calculer les distances entre les différentes features. Finalement, une lettre est reconnue lorsque sa distance au prototype de cette lettre est la plus faible parmi toutes les autres distances.

De plus, il est nécessaire pour pouvoir entraîner correctement l'outil, de posséder une base de données d'entraînement importante. Ce type de base de données n'est pas fréquent ou directement accessible en open-source. Tesseract use d'une base de données d'entraînement de 60160 exemples pour 94 caractères en plusieurs polices.

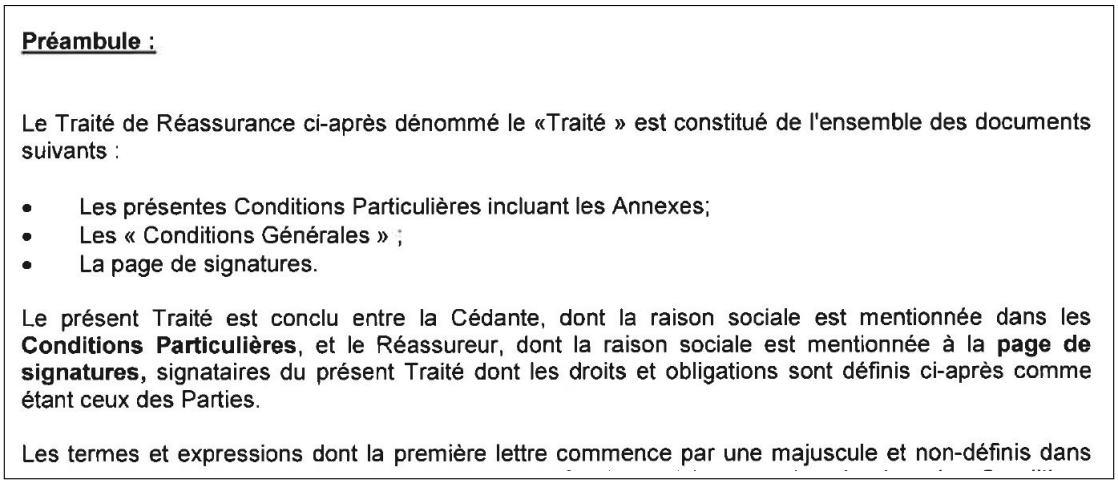
Une analyse linguistique est ensuite menée sur chaque séquence de lettres afin de choisir le mot du dictionnaire le plus proche. La notion de distance entre deux mots est ici donnée par le nombre de lettres à remplacer, à ajouter ou à retirer pour obtenir le mot du dictionnaire.

Afin d'obtenir de meilleur résultats, on peut renforcer notre base d'entraînement en utilisant une classifieur adaptatif. Il s'agit du même classifieur vu précédemment auxquels on ajoute à la base d'entraînement, les mots que l'on a classifié avec une grande confiance au fur et à mesure que l'outil les reconnaît.

27. Tesseract a par exemple constitué une base de prototypes comprenant plus de 60000 lettres

2.3.8 Résultats et applications

L'article [31] fait une étude de la précision des algorithmes reprenant toutes les étpes décrites plus hauts. Tesseract, par exemple, obtient des scores de l'ordre de 95 à 99% de précision dans la reconnaissance des mots ou des lettres. L'écriture tapuscrite est donc bien reconnaissable si les algorithmes utilisés sont adéquats et structurés d'une manière appropriée.
On peut également constater les résultats suivant obtenus sur le scan des conditions particulières d'un contrat de prévoyance d'une mutuelle santé :



Préambule :

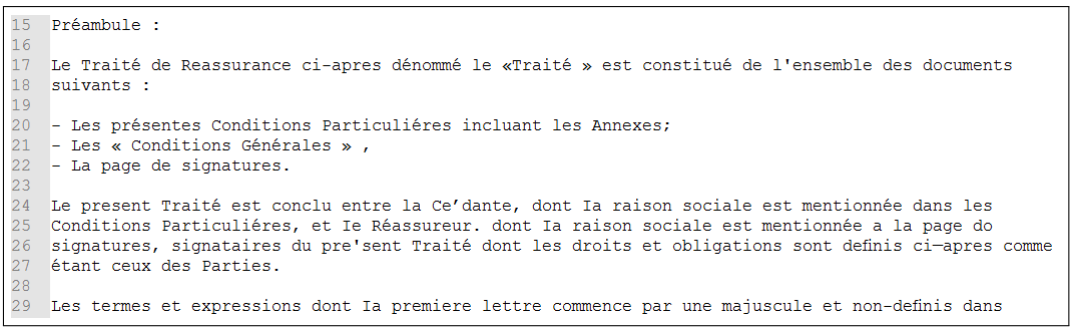
Le Traité de Réassurance ci-après dénommé le «Traité » est constitué de l'ensemble des documents suivants :

- Les présentes Conditions Particulières incluant les Annexes;
- Les « Conditions Générales » ;
- La page de signatures.

Le présent Traité est conclu entre la Cédante, dont la raison sociale est mentionnée dans les **Conditions Particulières**, et le Réassureur, dont la raison sociale est mentionnée à la **page de signatures,** signataires du présent Traité dont les droits et obligations sont définis ci-après comme étant ceux des Parties.

Les termes et expressions dont la première lettre commence par une majuscule et non-définis dans

(a) Image initiale



```
15  Préambule :
16
17  Le Traité de Reassurance ci-apres dénommé le «Traité » est constitué de l'ensemble des documents
18  suivants :
19
20  - Les présentes Conditions Particuliéres incluant les Annexes;
21  - Les « Conditions Générales » ,
22  - La page de signatures.
23
24  Le present Traité est conclu entre la Ce'dante, dont Ia raison sociale est mentionnée dans les
25  Conditions Particuliéres, et Ie Réassureur. dont Ia raison sociale est mentionnée a la page do
26  signatures, signataires du pre'sent Traité dont les droits et obligations sont definis ci—apres comme
27  étant ceux des Parties.
28
29  Les termes et expressions dont Ia premiere lettre commence par une majuscule et non-definis dans
```

(b) Texte reconnu

Le texte est intégralement reconnu, quelques petites erreurs dues aux accents ou autres ponctuations sont présentes. Ces résultats sont satisfaisants.

Nous décrivons dans la partie 4.1 un cas de nécessité d'un tel algorithme. De plus, dans le cas d'un document contenant un mélange d'écriture tapuscrite et manuscrite, cet algorithme peut être utilisé pour reconnaître la partie tapuscrite et la retirer afin de ne conserver que la partie manuscrite. De manière plus général, il est toujours plus facile de travailler sur un document où la sélection, la recherche de mot, le surlignage sont possibles. Nous avons donc développé un module qui transforme un fichier PDF composé de scans en un fichier PDF classique (où la sélection est possible).

Par ailleurs, si on utilise la méthode que nous venons de décrire sur de l'écriture manuscrite, les résultats sont extrêmement mauvais ; aucune lettre n'est correctement reconnue, même le découpage des mots n'est pas correct. Cela est du à l'ensemble des étapes que l'on a explicitées qui suppose une régularité de l'écriture. Aucune de ces étapes n'est à priori applicable à du texte manuscrit et notamment l'étape du matching des features. La prochaine partie va donc spécifiquement s'intéresser à de l'écriture manuscrite.

3 Les enjeux et les méthodes de la reconnaissance de l'écriture manuscrite

Même si l'écriture tend aujourd'hui à devenir de plus en plus informatique, il existe encore de nombreux documents contenant de l'écriture manuscrite. Dans un certains nombre de cas, l'écriture manuscrite reste un moyen de contact privilégié. Par exemple, les relations clients-entreprises se font encore assez souvent par l'intermédiaire de questionnaires à réponse manuscrite. Les certificats médicaux sont souvent écrits à la main. Automatiser la lecture de ces réponses peut être bénéfique pour de nombreuses administrations et entreprises ayant des relations client importantes. Certaines ont développé des technologies spécialisées pour certains de leurs besoins, c'est le cas des banques qui peuvent désormais lire les chèques de manière automatique, mais ces techniques ne se sont pas largement répandues dans le secteur de l'assurance pour le moment.

Dans cette partie, nous allons nous intéresser à la reconnaissance de l'écriture manuscrite et implémenter nos propres modèles. Contrairement à l'écriture tapuscrite, le problème de la reconnaissance de l'écriture manuscrite n'est pas encore bien résolu. Nous allons donc proposer une méthode basée sur la Data-Science et le Deep Learning. Pour fonctionner, l'algorithme doit avoir une architecture qui prend en compte les différents formes de documents où de l'écriture manuscrite est présente. Nous verrons également certains pré-traitements que l'on applique à l'image avant d'en extraire le texte, certaines méthodes vues dans l'écriture tapuscrite peuvent être réutilisées, par exemple dans la détection de lignes.

3.1 Différence avec l'écriture tapuscrite

Si le résultat recherché est le même pour l'écriture manuscrite et l'écriture tapuscrite, la donnée initiale n'est pas la même. Il s'agit toujours d'une image où le mot est écrit mais la forme de l'écriture manuscrite est beaucoup plus aléatoire. L'écriture manuscrite a une forme qui n'est ni régulière entre les auteurs ni régulière pour un même auteur dans un même document. De plus, dans l'écriture manuscrite les lettres se chevauche assez souvent ce qui est plus rare dans l'écriture tapuscrite. Cette grande variabilité et cette possibilité d'entrecroisement rendent le problème de la reconnaissance de l'écriture manuscrite beaucoup plus complexe. C'est pourquoi comme nous l'avons mentionné plus tôt dans ce mémoire, les outils utilisés ne peuvent être les mêmes pour l'écriture manuscrite et l'écriture tapuscrite.

3.2 Étape de la reconnaissance manuscrite

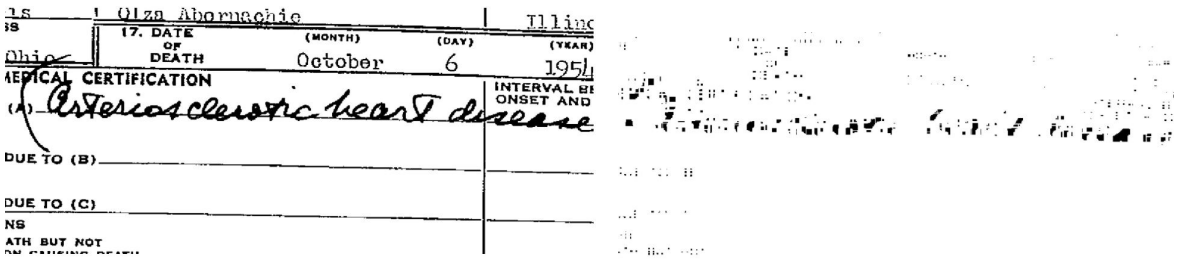
Comme pour l'écriture tapuscrite, une algorithme reconnaissant l'écriture manuscrite doit être structuré en étapes. La première étape consiste à segmenter la page ou le document afin de n'en conserver que les parties manuscrites. Dans un second temps, les parties manuscrites sont segmentés en image de mot puis finalement chaque image de mot est reconnue. On propose donc les étapes suivantes :

1. La première étape est la détection de zone de texte manuscrit
2. Ensuite, cette zone de texte va être segmentée en lignes et en mots
3. Finalement la reconnaissance des mots grâce à des modèles séquentiels

3.3 Détection de la zone de texte manuscrite

Dans un constat amiable par exemple, les zones de textes manuscrits sont souvent entourés de zones de textes tapuscrits. Il faut donc dans un premier temps les détecter automatiquement. Cette première détection peut se faire en utilisant une modélisation de l'épaisseur de l'écriture. De manière générale, l'épaisseur de l'écriture manuscrite est plus importante que celle de l'écriture tapuscrite. Une preuve en a été donné par Cheriet dans son livre [6]. L'idée pour extraire le texte manuscrit va donc être de retirer les pixels noirs provenant de traits ayant une faible épaisseur et de conserver les pixels noirs restants (l'image étant en noir et blanc).

L'épaisseur d'un pixel noir au coordonnées x, y sur l'image est estimée par la distance minimale entre deux pixels blancs qui encadrent ce pixel dans les directions horizontales SW_H et verticales SW_V donc $SW(x, y) = \min(SW_H, SW_V)$. A près avoir calculé l'épaisseur de chaque pixel noir d'une image, on applique un seuil de conservation des pixels, tous les pixels ayant une épaisseur en deçà de ce seuil sont transformé en pixel blanc. Ce seuil est déterminé par une analyse de l'histogramme de l'épaisseur des pixels. Voici un résultat que l'on obtient en utilisant cette technique :



(c) Image initiale (d) Pixel détecté

FIGURE 22 – Détection de l'écriture manuscrite

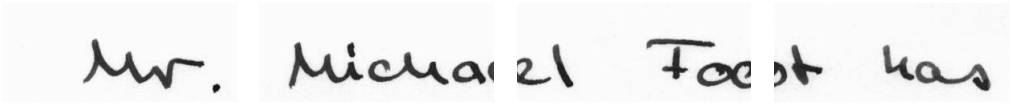
Le résultat est mitigé. Les pixels des zones de texte sont plus souvent conservés mais une détérioration du texte est évidente. Ainsi plusieurs pixels noirs provenant des tableaux et de l'écriture tapuscrite constituent du bruit et ne sont pas retirés tandis que certains pixels de l'écriture manuscrite l'ont été.

Afin de réduire le bruit, on peut ajouter une étape de retrait des lignes droites préalablement à l'application de la méthode. Les lignes droites ne proviennent pas d'un texte manuscrit de manière générale. La méthode utilisée est celle proposée et implémentée par les auteurs de cet article [35] qui s'appelle *Ligne Segment Detector*. Cette méthode se base sur les changements brutaux des niveaux de gris pour détecter des lignes droites et des segments sur une image. Après la détection et le retrait des pixels noirs provenant de lignes droites de l'image, les résultats obtenus sur notre exemple sont meilleurs mais toujours bruités.

Afin de diminuer encore le bruit, on utilise un outil de reconnaissance d'écriture tapuscrite pour détecter l'ensemble des lettres manuscrites sur le document. Les pixels noirs provenant des lettres tapuscrites reconnus avec une grande confiance sont ensuite retirés.

Finalement, afin de récupérer l'ensemble des pixels provenant des zones manuscrites, on définit des zones où beaucoup de pixels ont été conservés durant toute la procédure. Ces zones sont alors récupérés sur l'image initiale, ce qui nous donne l'écriture manuscrite complète. Dans le cas, où il y'a un entrecroisement entre les lignes droites, les lettres tapuscrites et l'écriture mansucrite (voir Fig 22), cette dernière étape est à éviter, ce qui induit une perte de la qua-

lité importante de l'écriture manuscrite, c'est une limite de notre approche. Dans un cas plus simple, cette procédure nous permet d'extraire des zones d'écriture manuscrite de ce type :



(a) Zone 1 (b) Zone 2 (c) Zone 3 (d) Zone 4

FIGURE 23 – Résultats finaux sur un nouvel exemple

3.4 Segmentation en lignes d'écritures et en mots

Les modèles que nous allons présentés plus loin prennent en entrée l'image d'un mot. Il faut donc préalablement détecter et découper l'ensemble des mots du documents à reconnaître.

La méthode proposée dans le cas de l'écriture tapuscrite se base sur l'utilisation des composantes connectées. Le calcul de ces composantes nécessite de la puissance de calcul. Calculez les composantes connectés dans le seul but de segmenter en lignes n'est pas efficace. Nous proposons donc une autre méthode.

Pour segmenter un texte selon les lignes d'écritures, il suffit de distinguer le nombre de pixels noirs par lignes. Dans les lignes de saut, ce nombre est faible tandis qu'il est plus élevé dans les lignes de texte.

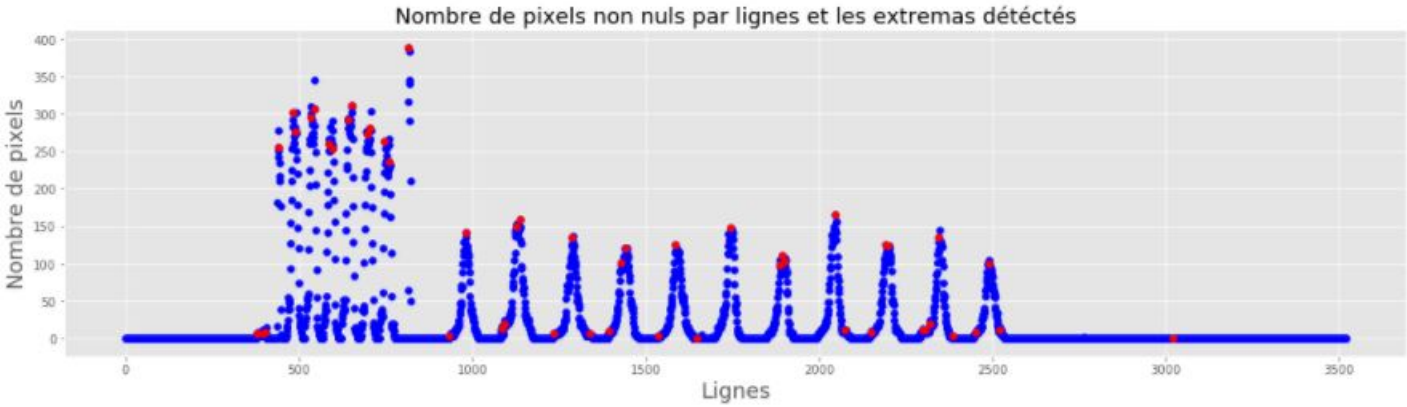
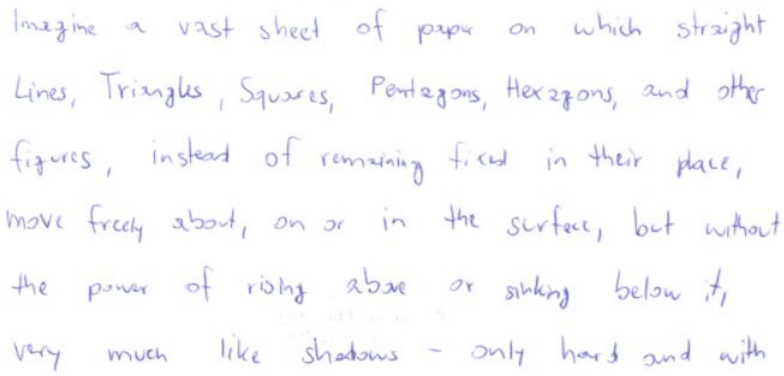
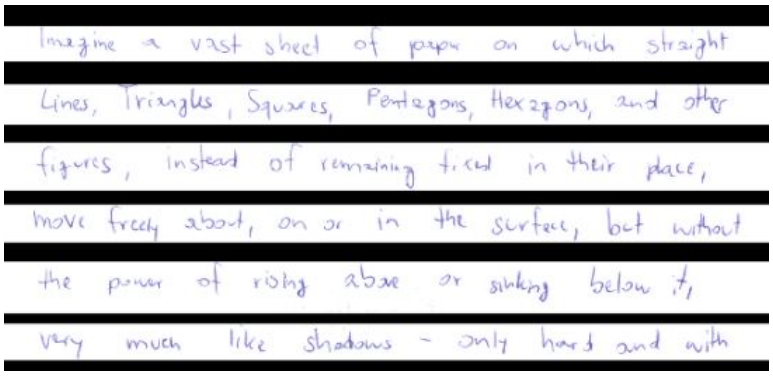


FIGURE 24 – Résultats obtenus en travaillant sur l'image 43

Pour segmenter de manière optimale et déterminer un seuil qui détermine si une ligne est écrite ou pas, on va utiliser un modèle de mélange gaussien. On suppose ainsi que le nombre de pixels noirs par ligne de texte suit une certaine distribution gaussienne tandis que le nombre de pixels noirs par ligne de saut suit une autre distribution gaussienne. Le modèle de mélange gaussien va estimer les paramètres de ces deux lois, puis déterminera selon quelle loi à été distribuée le nombre de pixels noirs pour chaque ligne. Cela impliquera pour chaque ligne sa désignation en ligne de texte ou ligne de saut.
Le résultat obtenu par cette modélisation est visible sur la Fig.25



(a) Image initiale



(b) Ligne retrouvé

FIGURE 25 – Segmentation en lignes

A la fin de la segmentation en utilisant le mélange gaussien, on a utilisé un lissage pour éviter que des lignes soient isolées ce qui perturbait la segmentation. Ce lissage consiste en une opération de convolution discrète entre le vecteur binaire u indiquant si une ligne est une ligne de texte ou une ligne de saut et un vecteur noyau K. Soit u le vecteur obtenu par la prédiction sur toutes les lignes en utilisant les modèle de mélange et K le noyau de convolution choisi de taille 20 pour assurer une zone de texte d'au moins à peu près 15 lignes et uniforme (même valeurs pour l'ensemble des coefficients du vecteur), on calcule

$$v = u \circledast K$$

où v est le vecteur résultant de l'opération de convolution. La convolution est appliquée sur les bords en ajoutant des 0, ce choix est justifié car généralement les bords d'une image ne contiennent pas de texte. Le vecteur v est seuillé à 0.5 pour donner un vecteur binaire qui est le résultat final de la segmentation en ligne.

La même méthode peut être utilisée pour segmenter une ligne en mots. On essaye cependant la modélisation d'une autre valeur que le nombre de pixels noirs par colonnes. Cette fois-ci c'est l'écart entre deux pixels noirs successifs, dans la lignes qui compte le plus de pixels noirs, qui est modélisé. Cet écart est normalement grand entre les mots mais petit entre les lettres. On obtient le résultats suivant sur une ligne de texte manuscrite.

(a) Image initiale

(b) Mot retrouvé

FIGURE 26 – Segmentaion en mots

Une légère erreur a été commise, ce qui montre les limites d'une telle modélisation. L'amélioration de la segmentation peut se faire par exemple par l'apprentissage de la taille du noyau de convolution et des valeurs des coefficients. Cela dépasse le cadre de ce mémoire.

La méthode de segmentation peut être utilisé en cascade pour segmenter de plus en plus finement des blocs de texte. D'autres résultats sur des pages plus complexes sont visibles en annexe 81. Ce résultat est issu de la première étape de la segmentation en mots d'un constat amiable. On remarque que la segmentation n'est pas parfaite. En effet, le constat amiable est composé de lignes de tailles différentes. Certaines lignes sont des demi-lignes et la configuration de notre modèle implique que ces lignes ne sont pas toujours considéré comme des lignes à part entière. Il faudrait donc pour améliorer les résultats permettre la possibilité des demi-lignes, de quart de ligne et donc analyser le document quart par quart.

3.5 Reconnaissance par HMM

Une fois les mots manuscrits détectés, il s'agit de les reconnaître. Reconnaître une lettre ou un nombre manuscrit unique et isolé est une tâche bien résolue aujourd'hui. Des réseaux de neurones bien entraînés sur de grandes bases d'exemples peuvent obtenir des scores pouvant atteindre 99% de réussite. Par exemple, sur la base de données MNIST, les résultats obtenus sont excellents [16].

Par contre, reconnaître une lettre ou un nombre qui s'inscrit dans un mot ou un chiffre est plus complexe. Un première possibilité est de définir pour chaque mot du vocabulaire une classe puis définir et entraîner un classifieur multi-classes pour classifier les images. Cependant, classifier l'image d'un mot entièrement, de manière automatique, n'est absolument pas optimal. Étant donné le grands nombre des mots existants dans la langue français, un classifieur à plus de 600.000 classes est extrêmement compliqué à mettre en oeuvre. De plus les bases de données pour entraîner un tel classifieur n'existent tout simplement pas. En effet, un tel classifieur nécessiterait un base de données contenant plusieurs images de touts les mots écrits à la main du vocabulaire français.

Reconnaître les lettres les unes après les autres est beaucoup plus simple. Ainsi une étape de segmentation en lettre peut être utilisée dans le traitement de l'image du mot. Cela permettrait de décomposer un mot en ses différentes lettres puis de reconnaître séparément les lettres. Mais étant donné la possibilité qu'ont les lettres manuscrites de s'entrecroiser, cette étape peut induire un nombre d'erreurs important dans la reconnaissance. Conscient de l'inefficacité d'une segmentation en lettres, une segmentation en *graphèmes* a été proposée dans la thèse de Xavier Dupré [8]. Un graphème est une petite partie d'une lettre à l'image d'un phonème dans les signaux audio.
Le découpage en graphèmes consiste en la détection des parties "ascendantes" et des parties

"descendantes" sur le squelette de l'image, toute forme en "u" est supposée être la frontière entre deux lettres. Ce découpage fournit ainsi des petites images appelés graphèmes (voir Fig.27).

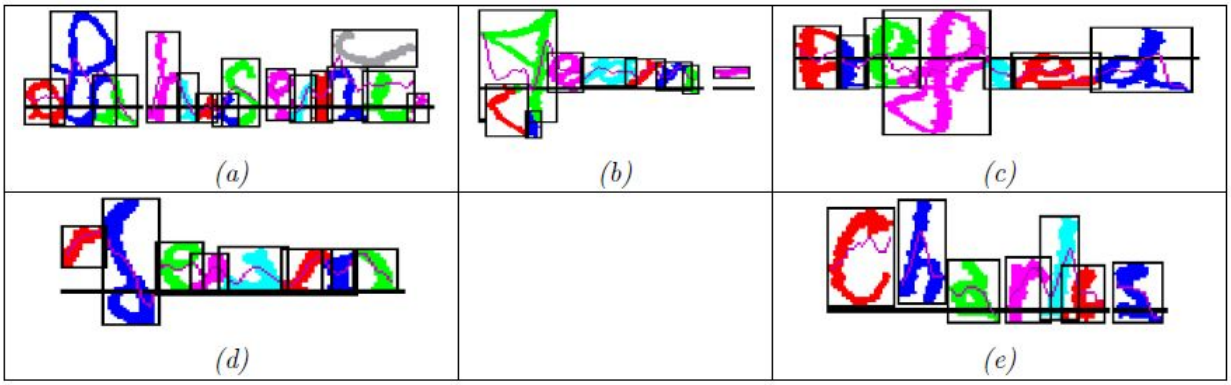


FIGURE 27 – Découpage en graphème [8]

L'image d'un mot est d'abord *nettoyé* pour retirer tout les pixels noirs isolés et les grands traits horizontaux et verticaux peu susceptibles d'appartenir à une lettre. Cette étape est primordiale pour assurer un découpage consistant.

Ces graphèmes vont alors être décrits par un vecteur caractéristique qui contient des informations tel que l'aire, la taille, l'espace entre les points du graphème et tout autre variable caractérisant la forme et la taille du graphème. Ces vecteurs caractéristiques vont ensuite être modélisés.

La modélisation des graphèmes la plus simple est constituée d'une classification des graphèmes puis d'une modélisation de la séquence des graphème reconnus. Cette méthode est utilisée depuis longtemps pour la reconnaissance de textes en écriture romaine et peut être élargie à d'autre langages [8]. Tout d'abord, on définit un nombre M de graphèmes possibles et on détermine pour chaque image de graphème, quel est le graphème reconnu parmi les M possibles. Cela est réalisé grâce à un classifieur. Les classifieurs peuvent être de toutes sortes (séparateurs linéaires, centres mobiles, réseau de neurones, arbre de décision, SVM...). On obtient alors une séquence de classes que l'on va ensuite modéliser par un modèle *Hidden Markov Model* (HMM). Le modèle HMM est expliqué en annexe 7.2 et un aperçu de l'aspect théorique du modèle est donné.

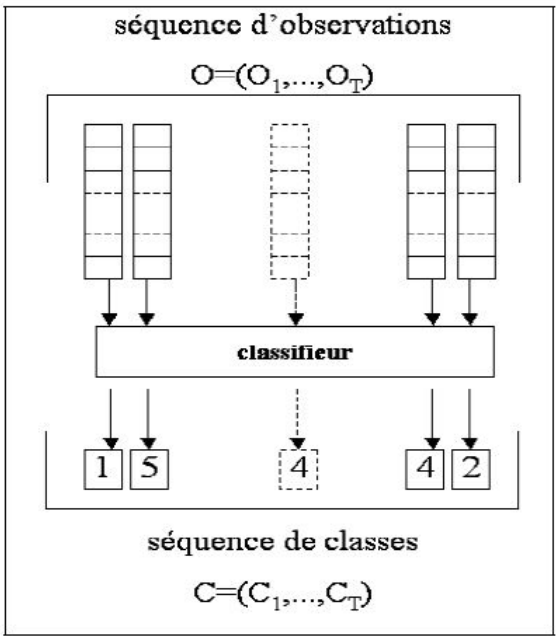


FIGURE 28 – Modèle de Markov Caché + Classifieur [8]

L'idée à l'origine de l'utilisation des HMM dans la reconnaissance d'écriture est que l'écriture manuscrite peut être considérée comme une séquence de gauche à droite de signaux un peu analogues aux signaux acoustiques. On va considérer une modélisation séquentielle de la suite des graphèmes car l'indépendance entre les graphèmes est une hypothèse qui n'est pas tenable. En effet, si l'on voit par exemple le premier graphème de la lettre G, il est fort probable de voir un deuxième graphème de G qui suit. L'idée est donc d'utiliser les chaînes de Markov à travers la modélisation HMM qui ne suppose pas l'indépendance des graphèmes entre eux pour modéliser la séquence de graphèmes. Précisons qu'une lettre peut être constituée de graphèmes de toutes formes et de tout types.

Cependant, il n'est pas évident qu'un processus (ou séquence de variables aléatoires) suive la loi d'une chaîne de Markov, c'est à dire que conditionnellement au "présent" le "passé" et le "futur" sont indépendants. Néanmoins, ce processus peut parfois être expliqué par un autre processus appelé processus caché, qui lui suit la loi d'une chaîne de Markov. Ce second processus est dit caché car le premier, celui qu'il explique, est le seul observé. In fine, le modèle HMM permet de convertir cette séquence observée en une probabilité : celle que le modèle émette cette séquence discrète.

Si l'on considère chaque lettre comme une association de graphèmes, on peut définir un modèle HMM pour chaque lettre. La composition de différentes modèles de lettres défini un modèle pour chaque mot. Finalement, on calcul la probabilité d'émission de la séquence observée (du mot à reconnaître) pour chaque modèle de mot. Le mot associé au modèle engendrant la plus grande probabilité est considéré comme reconnu dans la séquence de graphème. Voici une illustration du fonctionnement du modèle :

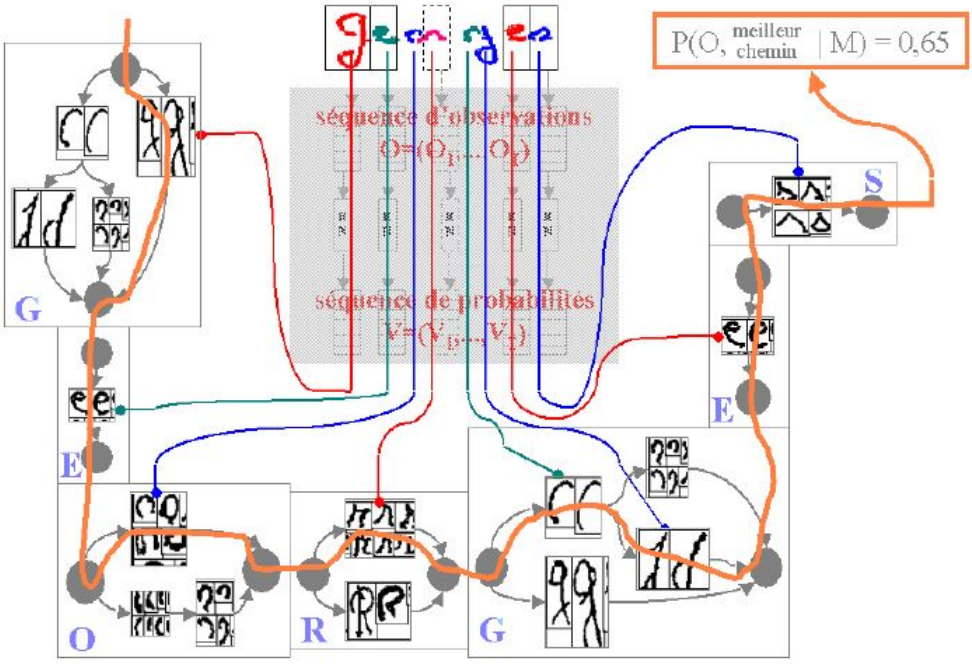


FIGURE 29 – Décryptage d'un mot à l'aide d'un modèle hybride réseau de neurones et modèle de Markov caché. Le réseau de neurones permet de reconnaître en classant chaque graphème parmi la centaine de classes disponibles. Le modèle de Markov caché associé au mot "Georges" résulte de la juxtaposition des modèles associés aux lettres qui le composent. Chacun d'entre eux est illustré par les séquences de classes de graphèmes les plus courantes qui permettent d'écrire une lettre. La ligne parcourant successivement les modèles des lettres du mot "Georges" correspond à la meilleure association entre la séquence d'observations extraites de l'image et la juxtaposition de celles apprises par chacune des lettres. [8]

Étudions le modèle par lettre c'est à dire le simple modèle HMM qui modélise la séquence de graphèmes associée à une lettre. Soit un *mot mathématique* noté $O = (O_1, ..., O_T)$ qui est la suite de vecteur caractéristiques des graphèmes et la séquence des classes d'observations associée à ce mot $C = (C_1, ..., C_T)$ (annotation), on définit la probabilité que le modèle M émette la séquence O par :

$$\begin{aligned} P(O|M) &= \sum_{q_1,...,q_T} P(O, q_1, ..., q_T|M) \\ &= \sum_{q_1,...,q_T} \left(P(q_1|M)P(C_1|q_1, M) \prod_{t=2}^{T} P(q_t|q_{t-1}, M)P(C_t|q_t, M) \right) \end{aligned}$$

Cette équation correspond à l'équation 1 présentée en annexe. Les valeurs $q_1, ..., q_t$ représente les états cachés du modèle HMM. Le remplacement de O par C dans l'équation correspond à la prédiction par le classifieur de la classe de chaque graphème. On a bien remplacé le mot mathématique O par sa classe C dans l'équation. Ce calcul est bien réalisable comme on l'a montré en annexe par l'algorithme *forward-backward*. La distribution $P(C_1|q_1, M)$ est supposée multinomiale puisque le nombre de classes C_1 est fini.

Pour chaque modèle de lettre, il y'a donc de nombreux paramètre à estimer : les valeurs de la matrice de transition, les paramètres de chaque loi multinomiale. Toutes ces estimations sont calculés par formules fermés présentés dans [8]. Une fois estimé, on peut représenter simplement le modèle HMM de chaque lettre par les classes et les connexions les plus fréquentes.

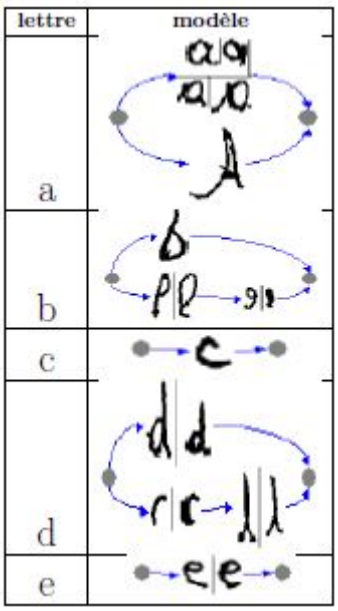


FIGURE 30 – Représentation simplifiée d'un modèle de Markov pour quelques lettres. Les points gris correspondent à un état en entrée et un état en sortie ajoutés au modèles. Chaque état émet une classe d'observations illustrée par un ou deux de ses éléments les plus probables. Chaque image de modèle résume les écritures les plus fréquentes pour chaque lettre. Ces lettres ont été estimées sur une base d'apprentissage de 30000 prénoms français. Seuls ont été conservés les transitions dont les probabilités sont supérieures à 0,15 afin de ne garder que les principaux coefficients sur les quelques centaines que contient chaque modèle. [8]

L'utilisation de modèles HMM présente encore quelques défauts. Le premier est la nécessité de segmenter l'image initiale en graphème ce qui peut introduire de l'aléatoire et donc des erreurs. Un second inconvénient du modèle HMM est que la modélisation suppose que chaque observation est dépendante seulement de l'état dans lequel se trouve la séquence caché. On ne modélise donc pas des interactions directement à court et à long terme entre les graphèmes.

Toutefois dans le cas de l'écriture manuscrite, où la donnée consiste en une trajectoire continue, la dépendance directe entre chaque observation est potentiellement importante. L'ajout de modèles externes pour prendre en compte cette dépendance peut être nécessaire. Toutefois augmenter le nombres de modèles fait croître exponentiellement le nombres de paramètres qu'il faut inférer et donc nécessite des bases de données d'entraînement plus larges pour entraîner les modèles. Tout ces défauts, et des avancées majeures dans le *Deep Learning* appliqué à la vision par ordinateur, ont entraîné le développement d'approches nouvelles dans la reconnaissance d'écriture manuscrite.

3.6 Reconnaissance par réseaux de neurones

Le Deep Learning ou *apprentissage profond* est devenu aujourd'hui la norme dans le traitement d'images. Popularisé par des résultats incroyables pour l'époque dans des concours de reconnaissance d'images (Imagnet [19]), le Deep Learning s'est peu à peu imposé comme un élément majeur du traitement de données non structurées comme l'image, le texte ou le son. Les premiers réseaux de neurones qui étaient des modèles ayant uniquement des couches de neurones simples (*Feedforward network*) ont été remplacés par des réseaux de neurones où certaines couches sont des opérations de convolution (CNN, *Convolutionnal Neural Network* ou *réseau de neurones convolutionel*) ou encore par des réseaux donc la structure va prendre en compte un effet temporel dans le cas par exemple de données séquentielles (RNN, *Recurrent*

Neural Network ou *réseau de neurones récurrent*).

Les premiers réseaux de neurones utilisés dans le cadre de la reconnaissance d'écriture manuscrite l'ont été pour appuyer les HMMs ([3], [1]). Les modèles HMMs hybrides ont ainsi utilisé les réseaux de neurones à des fins de classification simple de segments d'images (graphèmes) puis les modèles HMMs vont modéliser la séquence des classes reconnues (voir Fig.40). Toutefois ce type de modélisation ne lève pas les problèmes dont nous avons discutés plus haut étant donné que la modélisation de l'aspect séquentielle de l'image se fait toujours par des HMMs.

3.6.1 Introduction au Deep Learning et au réseau de neurones

Le *Deep Learning* est devenu la norme actuelle en termes de traitement de signaux et plus particulièrement dans le traitement d'images. Les réseaux de neurones qui constituent les algorithmes utilisés en Deep Learning sont des algorithmes ayant la possibilité de détecter des relations non-linéaires entre les données et de les exploiter à des fins de résolution d'une problématique. Cela n'est pas forcément possible pour d'autres algorithmes de machine learning qui se limitent à la mise en avant de relations de linéarité entre des données et un label.

Le Deep Learning est souvent utilisé dans des problèmes de reconnaissance d'images, de reconnaissance audio ou encore dans les moteurs de recherche. Le secteur industriel comme le secteur académique ont pris la mesure du potentiel de cette avancée scientifique et ont ainsi orienté leurs recherches sur la création et le développement de réseaux de neurones. De nombreuses universités développent ainsi des outils informatiques open-source où les principaux modèles du Deep Learning sont utilisables rapidement et efficacement.

Une large variété de réseau de neurones existe et ceux-ci sont adaptés à différents problèmes : la vision par ordinateur pour les réseaux de neurones convolutionels, les réseaux de neurones récurrents pour les problèmes séquentiels comme la reconnaissance audio ou l'analyse de fréquence boursière, ou encore des réseaux de neurones auto-encoder ayant la capacité de chercher une représentation optimale des données à la manière d'une *analyse en composante principale* (ACP) mais en essayant de résoudre un problème de manière supervisée contrairement à l'ACP qui se fait de manière non supervisée.

Tous ces réseaux ont été à la base conçus pour imiter le comportement du cerveau humain. En effet, la philosophie initiale du Deep Learning est basée sur le principe du *connexionnisme* : alors qu'un simple neurone biologique ou une seule variable dans un modèle de machine learning n'est pas intelligente, une large population de ces neurones peut en travaillant ensemble avoir des capacités cognitives importantes.

Toutefois, si l'idée générale est compréhensible, les raisons mathématiques de la réussite des réseaux de neurones ne sont pas clairement établies. Dans son article [22] et plus largement dans ces recherches, Mallat analyse les propriétés mathématiques d'un réseau de convolutions ainsi que ses capacités de séparation et d'invariance et pose un cadre d'analyse des réseaux de neurones. Il établit notamment la définition mathématique et la conception géométrique des réseaux de neurones et avance des hypothèses sur le fonctionnement des réseaux. Toutefois, il note le manque de l'ensemble des théorèmes de représentation, d'approximation en grande dimension et d'étude de convergence pour expliquer les raisons du bon comportement des réseaux. Ce manque scientifique est et continuera d'être l'objet de nombreuses recherches scientifiques mais contribue encore à l'appréhension et à la considération des réseaux de neurones comme

des boites noires dont la réussite est incomprise.

Dans cette partie, nous nous intéresserons davantage à l'application des réseaux de neurones dans un problème de reconnaissance d'écritures manuscrite et nous mettrons en avant les raisons de leurs succès dans notre application. Dans nos recherches, nous utiliserons souvent le livre de Goodfellow et Bengio [11] pour décrire les réseaux de neurones.

3.6.2 Structure classique d'un réseau de neurones et notation

Le réseau de neurones le plus classique et le plus simple appelé *Feedforward network* aussi connu sous le nom de *Multilayer Perceptron* est constitué d'opérations mathématiques simples. Le but du réseau est d'approximer une certaine fonction f^* tel que le label y de la donnée x puisse s'écrire de la manière suivante

$$y = f^*(x).$$

Le Feedforward network approxime la fonction f^* par une fonction à paramètres $f^* = f(\bullet; \theta)$ où θ est le paramètre à optimiser.

Ce modèle est appelé feedforward puisque il n'existe pas de liens qui permettent de réintroduire l'output du réseau dans le réseau. Ce type de lien sera vu plus tard dans les réseaux de type récurent (RNN).

L'*input layer* qui est la couche d'entrée, prend comme *input* une variable $x \in \mathbb{R}^n$ (de dimension $n = 4$ sur la figure 31) qu'il va ensuite multiplier par une matrice de poids notés $W \in \mathbb{R}^{n,d}$ pour fournir cela aux neurones du *hidden layer*. Le hidden layer ou *couche cachée* constitue l'ensemble des valeurs obtenues après la transformation une première fois de l'input. Les valeurs du hidden layer sont donc égales à $h = x \times W + c$ avec $c \in \mathbb{R}^d$ une constante. La dimension d correspond à la profondeur de la couche. On dit que la couche cachée possède d neurones.

Ces valeurs sont alors *activées* par une fonction d'activation ϕ communément la fonction ReLu, Tanh (tangente hyperbolique) ou encore la fonction Sigmoide (voir Tab.1). On obtient alors $\phi(x \times W + c)$.

Table 1 – Fonctions d'activations classiques

Fonction	Définition	Espace d'arrivée
Tanh	$x \to \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$[-1, 1]$
Sigmoide, $\alpha \in \mathbb{R}^+$	$x \to \frac{1}{1+e^{-x}}$	$[0, 1]$
Arctan	$x \to arctan(x)$	$]-\frac{\pi}{2}, \frac{\pi}{2}[$
ReLu	$x \to \max(0, x)$	$\mathbb{R}^+$

Après l'activation, les sorties de l'hidden layer vont encore être sommées pondérement (multiplié par une matrice de poids) pour être fournies à l'*output layer*. L'output layer ou *couche de sortie* va alors activer ces valeurs par une fonction d'activation (souvent la fonction Sigmoide, pour obtenir une probabilité) qui donne les valeurs de sortie. En résumé, on a le *chemin forward* du réseau qui est caractérisé par les équations suivantes

$$h = x \times W + c$$
$$o = \phi'(W'\phi(h) + c')$$

avec o la valeur en sortie (*output*) du réseau avec $W' \in \mathbb{R}^{d \times p}$ et $c' \in \mathbb{R}^p$ et ϕ' la fonction d'activation de l'output layer. Dans le cas par exemple d'une classification à m classes, on va définir $p = m$ et chaque valeur o_i de l'output du réseau pour $i \in [1, p]$ correspond à la probabilité que l'input x ait pour label la classe i.

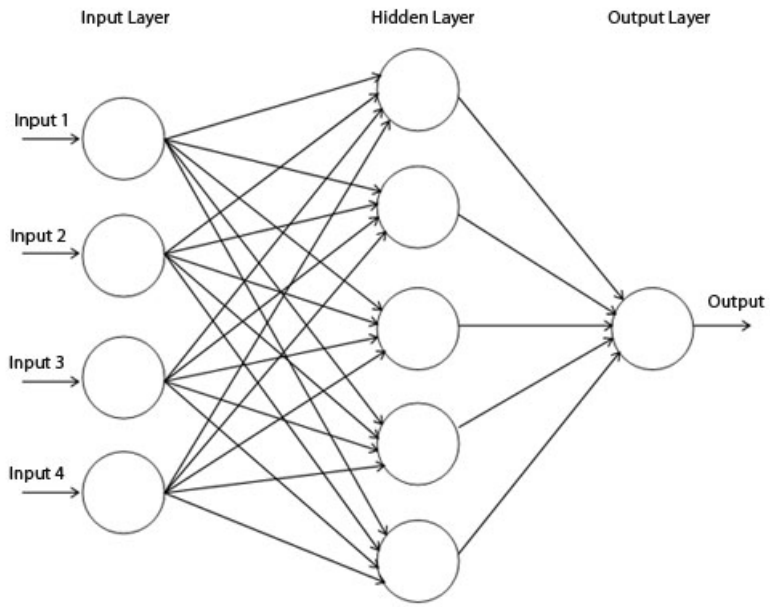


FIGURE 31 – MultiLayer Perceptron pour un problème de classification binaire, Synaptic

Dans ce réseau, les paramètres à optimiser seront l'ensemble des poids qui ont été utilisés dans les différentes couches W, W', c, c'. Plus le nombre de neurones par couche n, d, p est élevé, plus le nombre de paramètres est important.

3.6.3 Optimisation du réseau, fonction de perte et back-propagation

Ces paramètres sont optimisés grâce à une *fonction de perte* (parfois aussi appelés *fonction objectif*) qui analyse la valeur en sortie de l'output layer et la compare aux valeurs réelles y (fournis lors de la phase d'entraînement) et calcule une perte. Par exemple, si la fonction de perte est la fonction de perte des moindres carrées, la perte est calculée par la formule suivante :

$$J(\theta) = \frac{1}{4} \sum_{x \in \mathbb{X}} (y - \phi'(W'\phi(x \times W + c) + c'))^2$$

avec $\theta = W, W', c, c'$ et $\mathbb{X}$ l'ensemble des données observées.

Plus généralement, la plupart des réseau de neurones sont entraînés en maximisant la vraisemblance. Cela signifie simplement que la fonction de coût est la log vraisemblance négative également considérée comme la *cross-entropy* entre les données d'entraînement et la distribution du modèle. Ce cout est donnée par la fonction

$$J(\theta) = -\mathbb{E}_{x,y \sim \hat{p}_{data}} \log p_{\text{model}}(y|x).$$

Dans le cas d'une distribution gaussienne pour $p_{\text{model}}(y|x)$ on retrouve la fonction de perte des moindres carrés. Dans le cas d'une distribution de bernoulli, on trouve la *log-loss* définit par

$$-\log P(y|\hat{y}) = -(y \log(\phi'(W'\phi(x \times W + c) + c')) + (1 - y) \log(1 - \phi'(W'\phi(x \times W + c) + c')))$$

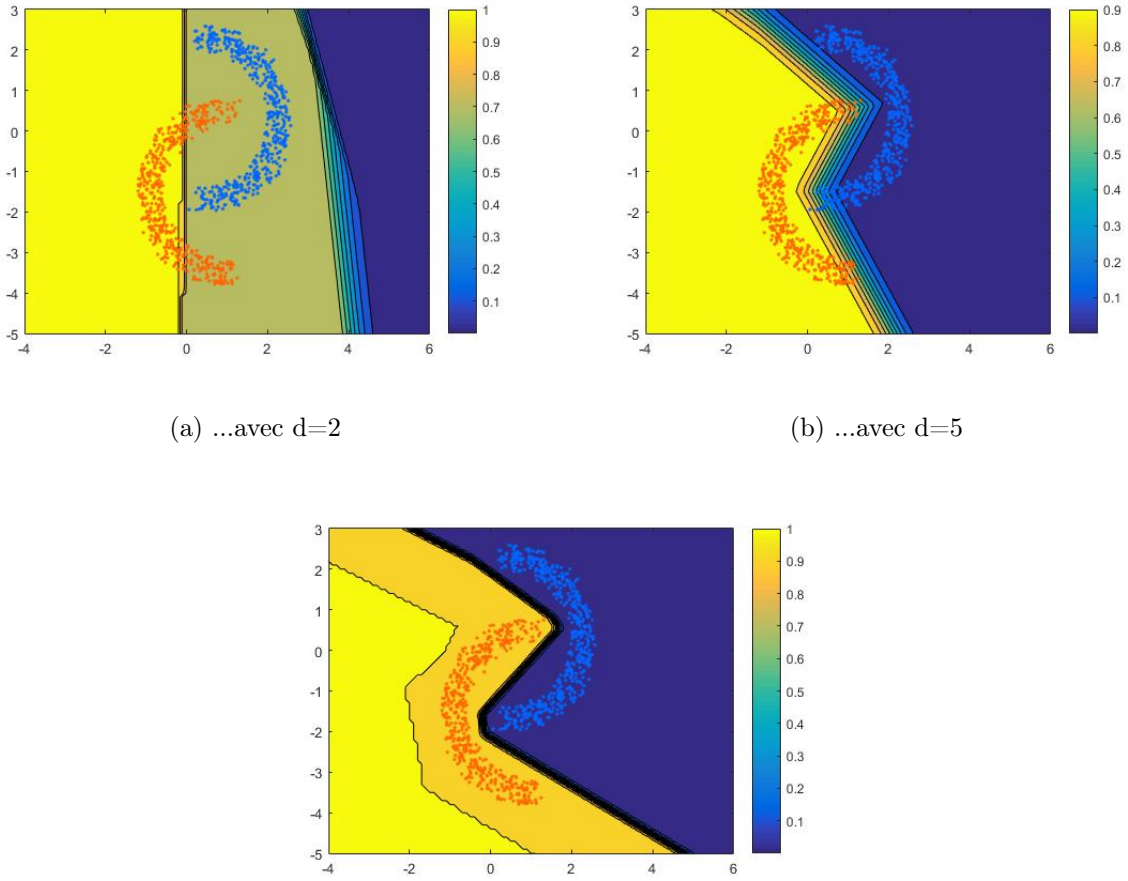
Cette fonction de perte permet de définir une *descente de gradient* (voir [32]) sur l'ensemble des poids du réseau, c'est ce que l'on appelle la Back-Propagation. La descente de gradient est un algorithme classique d'optimisation. L'algorithme définit aléatoirement un point de départ x puis va calculer à chaque étape le gradient de la fonction à optimiser pour modifier la valeur de x de manière à "aller" dans le sens opposé du gradient. Mathématiquement, si on note f la fonction à optimiser, on a à chaque étape de l'algorithme le calcul suivant

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k)$$

avec x_0 le point initial à choisir (aléatoirement ou pas) et $\nabla f(x_k)$ qui est le gradient de la fonction f au point x_k.

Un des paramètres importants de l'optimisation du réseau est alors le *learning_rate* qui est la valeur qui multiplie le gradient dans la descente de gradient (notée α_k dans l'équation). Cette valeur peut être constante dans le temps ou décroître. Si cette valeur est élevée, le réseau converge plus rapidement mais parfois trop rapidement et peut ne pas trouver l'extremum, une valeur trop faible implique une convergence plus lente mais parfois plus précise.

Le feedforward network va réussir à approximer la fonction f^* et ainsi dans un problème de classification, à plus ou moins bien classifier les données selon le degré de complexité et de profondeur du réseau (Voir Fig.32).



(a) ...avec d=2

(b) ...avec d=5

(c) ...avec d=10

FIGURE 32 – Résultats d'un feedforward network ...

La figure 32 montre la capacité du réseau de neurones à classifier en fonction de la profondeur (ou dimension) de la couche cachée.

Le réseau présenté plus haut est le plus simple parmi les réseaux de neurones mais ne peut être utilisé facilement dans le traitement d'images. En effet, si on considère une image de taille 1080 par 720, c'est à dire avec $1080 \times 720 = 777600$ pixels et qu'il y'ait quelques 10 neurones dans le hidden layer, cela porte le nombre de coefficients à optimiser à $777600 \times 10 \approx 8M$ de paramètres. Ce nombre est déjà très grand, d'autant plus qu'un réseau de neurones est souvent constitué de plusieurs hidden layer les unes à la suite des autres et que le nombre de neurones par hidden layer est important, bien plus grand que 10 normalement.

Ainsi, l'utilisation de réseaux de neurones feedforward pour travailler sur des images nécessite des réseaux de neurones de taille immense dont l'optimisation est très incertaine. Il faudrait des bases de données assez grandes pour que chaque paramètre soit correctement calculé. De plus des propriétés d'invariance à la translation et aux rotations sont nécessaires dans la reconnaissance sur images. Des réseaux plus adaptés comme le réseaux de neurones convolutionels sont utilisés.

3.6.3.1 Réseau de neurones convolutionels (CNN)

Pour travailler sur des images, les réseaux de neurones convolutionels (CNN) possèdent trois caractéristiques importantes :

- La parcimonie des interactions, c'est le fait que le CNN peut, par exemple, se focaliser sur une dizaine ou une centaine de pixels d'une image pour en extraire de l'information sans avoir à prendre en compte l'ensemble de l'image.
- Le partage des paramètres, c'est le fait que des filtres identiques sont appliqués à l'ensemble de l'image.
- L'invariance de la représentation, qui est en partie due aux partages des paramètres, est le fait que les informations extraites grâce au réseau ne dépendent pas de leurs positions dans l'image. Ainsi par exemple, une voiture peut être reconnue dans une image quelque soit sa position dans l'image.

Concrètement, les CNNs ont révolutionné l'approche de la *computer vision* (vision par ordinateur). Dans le passé, l'analyse des images se faisait grâce à l'extraction de features c'est à dire des points particuliers sur une image, puis dans un second temps ces features étaient classifiées. Les CNNs permettent de faire ces deux étapes conjointement tout en minimisant l'erreur de classification.

Voici une explication du fonctionnement des réseaux de neurones convolutionels et plus particulièrement de la couche de neurone qui implique le nom du réseau : la couche de *convolution*. La convolution est une opération mathématique couramment utilisée dans le traitement de signal. Elle permet par exemple de débruiter un signal ou de le concentrer autour d'un point ou encore de l'agréger à un autre signal. L'opération de convolution sur une image (qui n'est rien d'autre qu'une matrice de valeurs de pixels, voir 2.3.1), est l'application coulissante d'un filtre (matrice carrée, souvent de taille 2,3 ou 5) sur l'input du réseau. Cette application est une multiplication terme à terme entre les valeurs du filtre et les valeurs de l'image suivie d'une somme (voir Fig 48).

Notons K le filtre de la convolution appelé aussi noyau de convolution de taille $k \times k$ et X

la matrice représentant l'image de taille $n \times m$, la convolution est l'application

$$\phi_K : \mathbb{R}_{n\times m} \to \mathbb{R}_{n'\times m'}$$

qui va associer à la matrice X, une matrice X' de taille $n' \times m'$ dépendant du *stride*. Le stride p est le pas de l'application du noyau sur l'image. Ainsi, on a

$$n' = \frac{n-k}{p} + 1$$

et

$$m' = \frac{m-k}{p} + 1.$$

Finalement la matrice X' est défini par $\forall (i,j) \in [\![1, n']\!] \times [\![1, m']\!]$

$$X'_{i,j} = \sum_{l=i}^{i+k} \sum_{m=j}^{j+k} K_{l,m} X_{l,m} + c.$$

Les poids $K_{1,1}, K_{1,2}, ..., K_{k,k}, c$ sont les paramètres à apprendre.

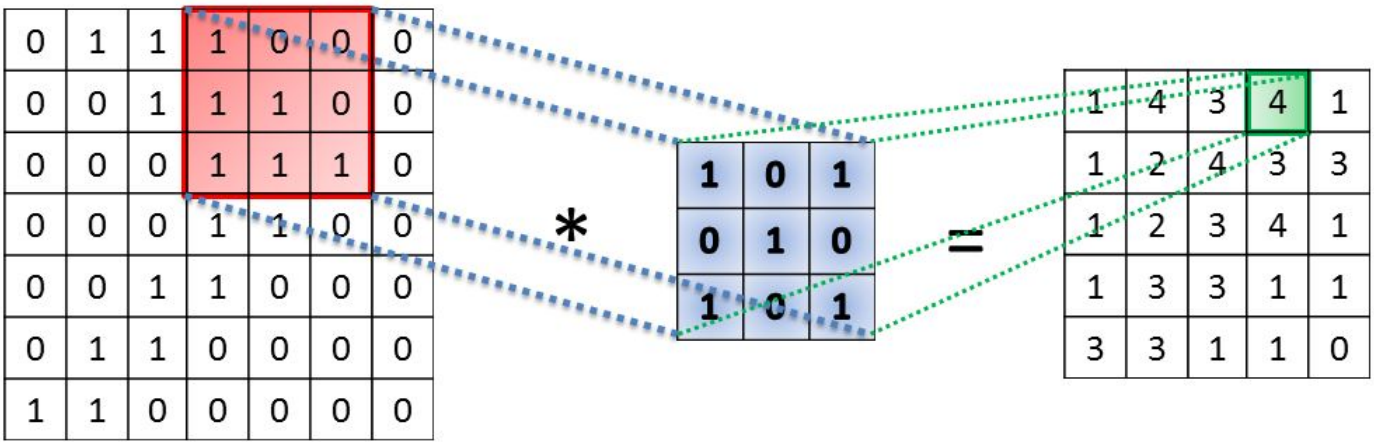


FIGURE 33 – Opération de convolution, kernel de taille 3, stride de 1

Ainsi l'application de la matrice K sur l'image se fait de manière coulissante sur l'image. Toutefois on peut décider de faire coulisser le noyau en sautant des cases à chaque fois, c'est le stride que l'on choisit, c'est à dire le nombre de cases que l'on saute entre deux applications du filtre (verticalement et horizontalement), un stride de 1 signifie que l'on ne saute pas de cases.

De manière pratique, l'opération de convolution nécessite une attention particulière pour les bords. Parfois, on impose la contrainte que le centre du noyau de convolution touche l'ensemble des pixels de l'image. Cela pose un problème au niveau des bords. Dans ce cas, on peut utiliser le *padding*, c'est à dire compléter les bords de 0 pour que la convolution se fasse. Cela revient à agrandir l'image en ajoutant des 0 sur tous les bords puis à appliquer la convolution normalement.

Après l'étape de convolution, une étape de *pooling* est très souvent réalisée. L'étape de pooling consiste en une sélection de valeur (max,min,moyenne,..) selon des fenêtres coulissantes à la matrice issue de la convolution (voir l'exemple du max pooling sur Fig.34). Le max pooling se traduit mathématiquement par le fonction maximum de plusieurs noyau de convolution. Notons $e_{1,1}, ..., e_{n,m}$ la base unitaire des matrices $\mathbb{R}_{n,m}$, le max pooling est l'application

$$\psi : \mathbb{R}_{n,m} \to \mathbb{R}_{n',m'}$$

avec

$$n' = \frac{n-k}{p} + 1$$

et

$$m' = \frac{m-k}{p} + 1$$

qui à X va associé

$$X' = \max_{(i,j)\in[\![1,n]\!]\times[\![1,m]\!]} \phi_{e_{i,j}(X)}$$

avec le max étant le maximum éléments par éléments d'une matrice. L'idée du max pooling est d'extraire les valeurs maximum d'un signal qui sont censées expliquer plus précisément la variabilité du signal. Le max pooling est aussi nommé *subsampling* chez certains auteurs.

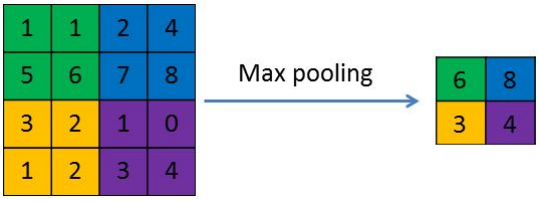


FIGURE 34 – Max Pooling, avec un kernel de taille 2 et stride de 2

Un réseau CNN possède en général plusieurs couches de convolutions et de max pooling les unes à la suite des autres avec des dimensions qui augmentent et se réduisent au fur et à mesure de l'avancée dans le réseau. Voici l'un des premiers réseaux de neurones convolutionels [20] construit :

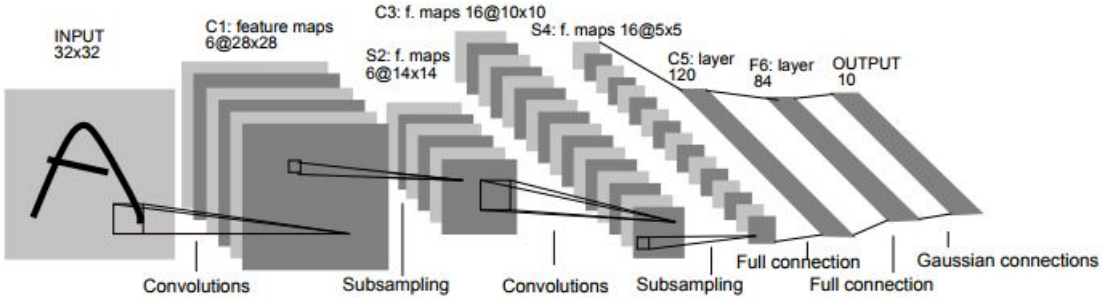


FIGURE 35 – Réseaux de neurones de convolutions, [20]

Ce réseau va simplement appliquer à l'image une suite de convolutions suivie d'étapes de max-pooling. Les dernières couches sont des *fully-connected layer* c'est à dire des couches de neurones ayant un comportement similaire à celui d'un hidden layer d'un feedforward network.

Ce réseau fut proposé en 1998, les meilleurs réseaux actuels comme AlexNet ou ResNet sont beaucoup plus développés, pouvant prendre en input une image de couleur et non plus seulement une image noir et blanc et ayant une architecture encore plus longue. De plus, les bases de données existantes sont de plus en plus importantes, AlexNet a par exemple été entraîné sur près d'un million d'images, ce qui assure la qualité de la reconnaissance.

A noter que ces réseaux vont répondre à des questions simples : est-ce qu'il y a une voiture dans l'image ? est-ce qu'il y a un avion dans l'image ? Parfois également quelle est la lettre (unique) dans l'image ? Dans le cas de la reconnaissance de l'écriture manuscrite, il s'agit de reconnaître sur l'image une séquence de lettres qui se suivent, cela est donc plus complexe.

3.6.3.2 Réseau de neurones récurrents (RNN)

Les réseaux de neurones récurrents sont une famille de réseaux de neurones pouvant travailler sur des données séquentielles. Ces réseaux comme les réseaux de neurones convolutionels ont la possibilité de travailler sur des inputs de tailles différentes (la longueur de la séquence pouvant différer). L'idée de ces réseaux est de permettre le partage des paramètres le long de la séquence afin de permettre l'extraction de l'information issue de l'aspect séquentiel de la donnée. La comparaison d'un RNN et d'un feedforward network donne une idée de l'apport de RNN par rapport au réseaux de neurones simples.

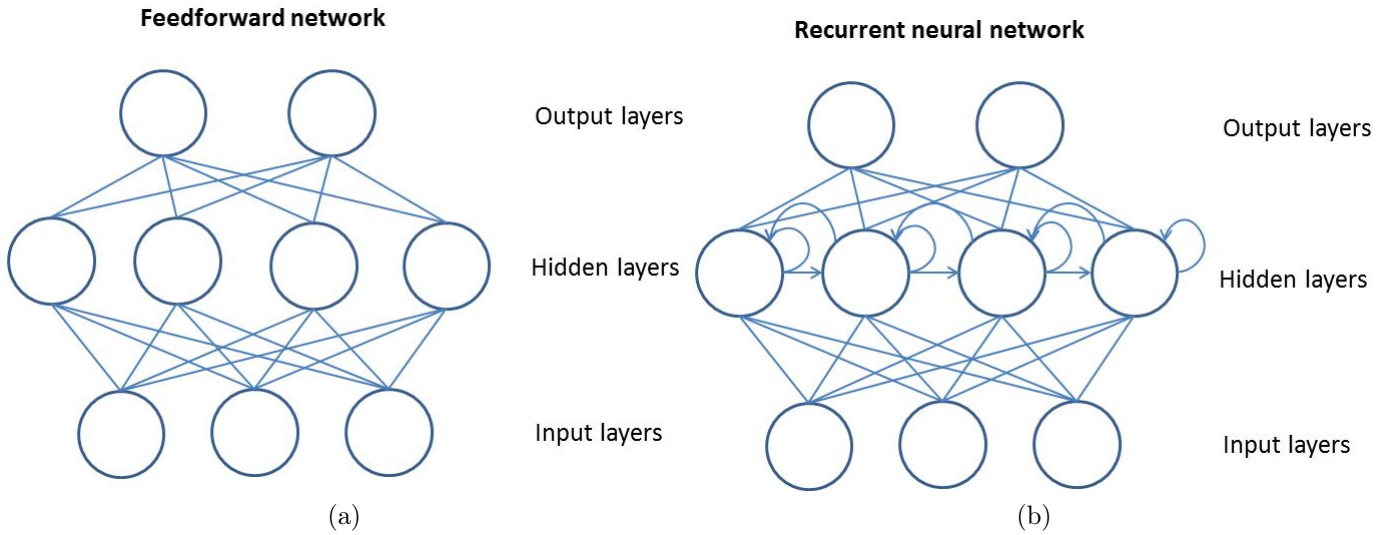


FIGURE 36 – Comparaison d'un feedforward network et d'un RNN

On peut voir sur la figure 36 que le RNN introduit des connections au sein même de la couche cachée. On peut même envisager des interactions entre les couches cachées où des couches plus avancées renverraient leurs outputs vers des couches moins avancées plus loin dans la séquence.

Ainsi si on note $x^{(1)}, ..., x^{(T)}$ la séquence en entré du réseau, on obtient la couche cachée $h^{(t)}$ par $h^{(t)} = f(h^{(t-1)}, x^{(t)}, \theta)$ avec θ le paramètre de la fonction f. La fonction f peut être la même à chaque temps t ce qui assure le transfert des paramètres au sein du réseau. De plus, la longueur de la séquence n'est pas imposée par le réseau puisque la fonction f pouvant être utilisée de manière récursive à l'infini.

Plus concrètement, la fonction f est souvent modélisée par une fonction linéaire comme le montre la figure 37.

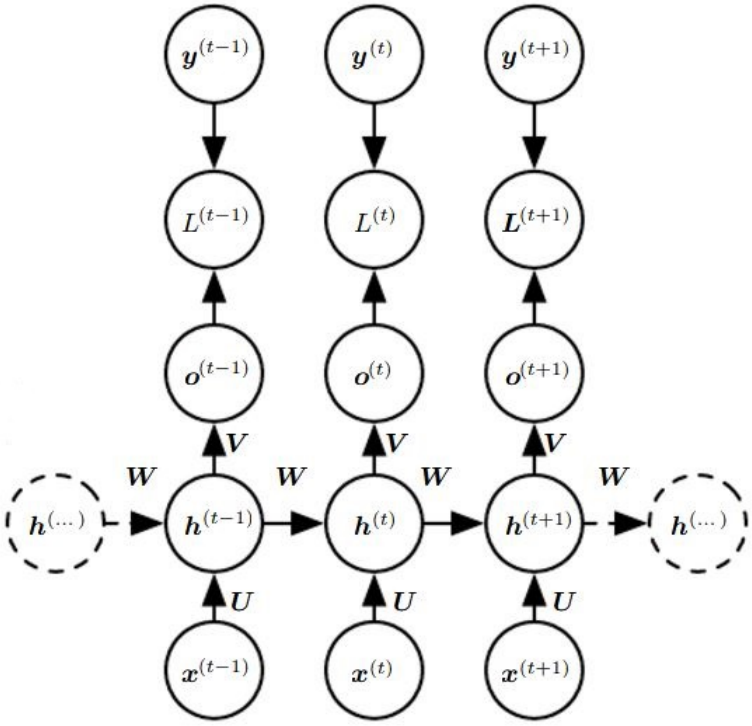


FIGURE 37 – Exemple de la manière d'entraîner un RNN. La fonction de perte est définie par L qui va comparer les sorties o du réseau de neurones et la vraie séquence y. Cette fonction de perte permet d'entraîner le réseau par back-propagation appelé *back-propagation through time.* [11]

Le RNN vu sur la Fig.37 a un chemin forward caractérisé par les équations suivantes :

$$
\begin{aligned}
a^{(t)} &= b + Wh^{(t-1)} + Ux^{(t)} \\
h^{(t)} &= tan(a^{(t)}) \\
o^{(t)} &= c + Vh^{(t)} \\
\hat{y}^{(t)} &= \text{softmax } o^{(t)}
\end{aligned}
$$

avec b, c, W, U, V les paramètres du réseau. Les valeurs $\hat{y}^{(t)}$ vont alors être comparées aux vrais valeurs $y^{(t)}$ pour définir une fonction de perte qui sera encore une fois minimisée par back-propagation. Cette back-propagation est appelée *back-propagation through time.*

Parfois, lors de la propagation de l'information le long de la séquence h, l'information se trouve diluée au fur et à mesure de son passage dans les différentes fonctions d'activation. Afin de conserver en mémoire à long terme l'information provenant du début de la séquence, plusieurs réseaux de neurones utilisent une architecture qui permet de ne pas modifier à chaque itération la valeur conservée de la séquence et en même temps à ne pas rencontrer le problème du "vanishing gradient problem".

Le "vanishing gradient problem" est un problème constaté lors de l'utilisation de la back-propagation pour l'entraînement de n'importe quel réseau de neurones. L'utilisation de la 'chain-rule' pour calculer le gradient des premières couches du réseau entraîne un affaiblissement du signal de l'erreur au fur et à mesure de sa propagation dans le réseau. Les fonctions d'activation tendent à éviter le problème en augmentant la puissance du signal.

Il existe deux types de couches dans les réseaux de neurones qui conservent l'information séquentielle à long-terme : la couche *LSTM* pour *Long short-term memory* et *GRU* pour *Gated recurrent unit*. Les deux types de couches ont des structures similaires avec le deuxième type ayant un peu moins de paramètres à apprendre.

Une couche LSTM dans un réseau de neurones est une séquence de neurones LSTM ayant tous les mêmes paramètres. Un neurone LSTM est lui même la combinaison de plusieurs neurones. En effet un neurone LSTM comprend une combinaison de portes (on peut les considérer comme des neurones) qui vont plus ou moins laisser passer de l'information. Cette combinaison est illustrée dans l'image 38 [28].

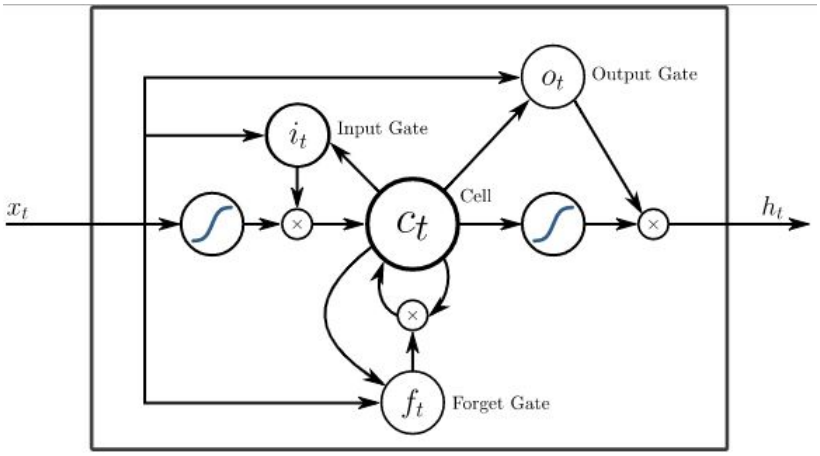


FIGURE 38 – Neurone LSTM, OKStateACM

Plus formellement, voici comment l'output du neurone h_t est calculé :

$$
\begin{aligned}
f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\
h_t &= o_t \circ \sigma_h(c_t)
\end{aligned}
$$

avec σ_g la fonction sigmoid, σ_c et σ_h la fonction tangente hyperbolique, tandis que l'opérateur $\circ$ est l'opérateur d'Hadamard.
Les trois "portes" f_t, i_t, o_t sont calculées comme une fonction linéaire du passé h_t et de l'input x_t et passe à travers la fonction Sigmoide qui transforme cela en probabilité. La "porte de l'input" i_t décide à quel niveau l'input a une action sur la mémoire, la "porte de l'oubli" f_t décide à quelle niveau l'information passée est utilisée pour calculer l'output du neurone et la "porte de sortie" o_t détermine à quel niveau la mémoire aura un impact sur la valeur de sortie du neurone h_t. Pour chaque porte, une valeur de 1 laisse passer toute l'information tandis qu'une valeur de 0 ne laisse rien passer.

La couche GRU est formellement définie ainsi :

$$
\begin{aligned}
z_t &= \sigma_g(W_z x_t + U_z h_{t-1} + b_z) \\
r_t &= \sigma_g(W_r x_t + U_r h_{t-1} + b_r) \\
h_t &= z_t \circ h_{t-1} + (1 - z_t) \circ \sigma_h(W_h x_t + U_h(r_t \circ h_{t-1}) + b_h)
\end{aligned}
$$

28. `https://github.com/OKStateACM/AI_Workshop/wiki/LSTM-Generator`

où les matrices W, U, b sont des paramètres du modèle.

Ces deux types de structures vont laisser plus ou moins passer l'information au cours de la séquence. Ainsi si une information doit être conservée dans le temps sans être perturbée par l'input plus récent, les portes f_t et o_t vont être largement ouvertes tandis que la porte i_t va être fermée.

Les réseaux de neurones RNN, LSTM (qui contiennent des couches LSTM) ou GRU peuvent être bidirectionnels, c'est à dire prendre en compte le sens forward et backward dans le temps. Cela peut être utile dans le cas de la reconnaissance de l'écriture manuscrite où les lettres du passé mais aussi les lettres futures peuvent aider à la prédiction de la lettre présente. En parallélisant deux couches de neurones récurrentes l'une allant dans un sens et l'autre allant dans le sens contraire puis en concaténant ou en sommant ces output, on crée facilement un réseau de neurones récurrents bidirectionnels.

Pour entraîner ce type de réseau de neurones, il faut pouvoir aligner chaque input x_t avec un label y_t, c'est à dire associer un label à chaque élément de la séquence d'input. Dans le cas de l'écriture manuscrite, il faut donc aligner chaque image de la lettre ou des features extraites de celle-ci (x_t) à la lettre représenté numériquement par une valeur (y_t). Pour obtenir les x_t il faut segmenter l'image et c'est précisément ce que nous voulons éviter. On va donc voir comment éviter l'étape de la segmentation et fournir directement l'image toute entière au réseau.

3.6.4 Application du Deep Learning à la reconnaissance de l'écriture manuscrite, output layer CTC

Dans ces articles et dans son livre ([12], [13] [14]), Alex Graves introduit une nouvelle architecture de réseau de neurones et plus particulièrement une nouvelle couche de sortie (*output layer*) de réseau de neurones appelée *Connectionnist Temporal Classifcation*, (CTC) qui lui permet de faire directement de la classification séquentielle sans passer par l'étape segmentation de l'input. Voici une brève explication de la nouvelle approche qu'il propose.

Au lieu de chercher à segmenter l'image en lettres puis à extraire de chaque partie de l'image des *features* qui permettent alors de faire de la classification et de retrouver les lettres, Graves propose d'utiliser directement l'image d'un mot comme input à un réseau de neurones récurrent et une séquence de lettres comme output pour l'entraînement du réseau. Chaque image de mot manuscrit est ainsi traitée pour en extraire une séquence $x = (x_1, ..., x_T)$, on peut prendre par exemple x_1 comme étant la première colonne de pixels de l'image et ainsi de suite. Le label de l'image est une séquence de chiffre qui symbolise une séquence de lettres, chaque chiffre correspondant à une lettre.

La possibilité de mettre en sortie du réseau de neurones directement la séquence de lettres à apprendre (qui n'a pas la même taille que l'input !) est rendue possible par l'utilisation de l'output layer CTC. Le fonctionnement de la couche de sortie du réseau est décrite en détail dans l'article [13].

Considérant l'ensemble A des caractères pouvant être reconnus, on définit l'ensemble $L = A \cup \{blanc\}$ où *blanc* est la possibilité pour le réseau de neurones d'indiquer qu'il ne reconnaît rien. L'output layer est ensuite défini comme une simple couche de sortie de réseaux de neurones activé par la fonction *Sigmoide* du type de l'output layer d'un feedforward network avec $\mid L \mid$

comme nombre de classes pouvant être reconnu et qui est appliqué à chaque instant t de la séquence. Le réseau de neurones va donc fournir en sortie la probabilité qu'à l'instant $t \in [1, T]$, la classe de sortie soit l'élément $k \in L$, notons cette probabilité y_k^t. On obtient une matrice de probabilité de taille $T \times L$. En voici un exemple,

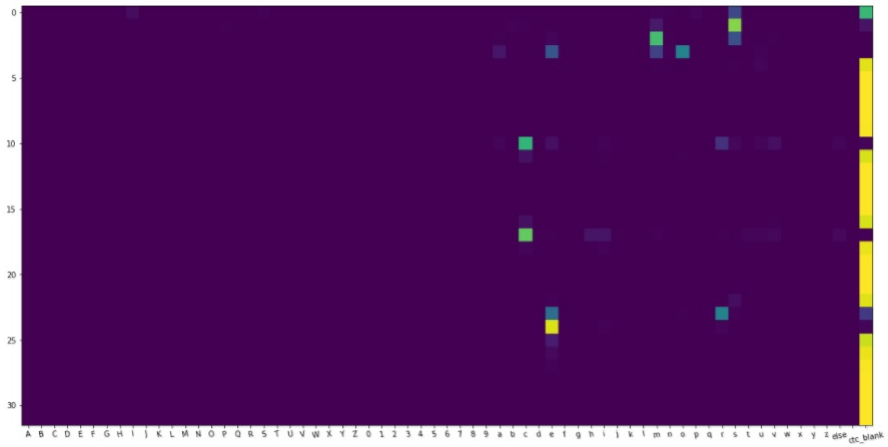


FIGURE 39 – Matrice de probabilité en sortie de l'output layer avec comme input une image de l'écriture manuscrite du mot 'marche'. En abscisse les classes, en ordonnée le temps, la couleur représente la valeur de la probabilité, la couleur violette correspondant à la valeur 0.

Voici un autre exemple de probabilités obtenues dans la reconnaissance de phonèmes dans le cadre d'un modèle de reconnaissance de la parole (*speech to text)* qui est un problème assez similaire au notre :

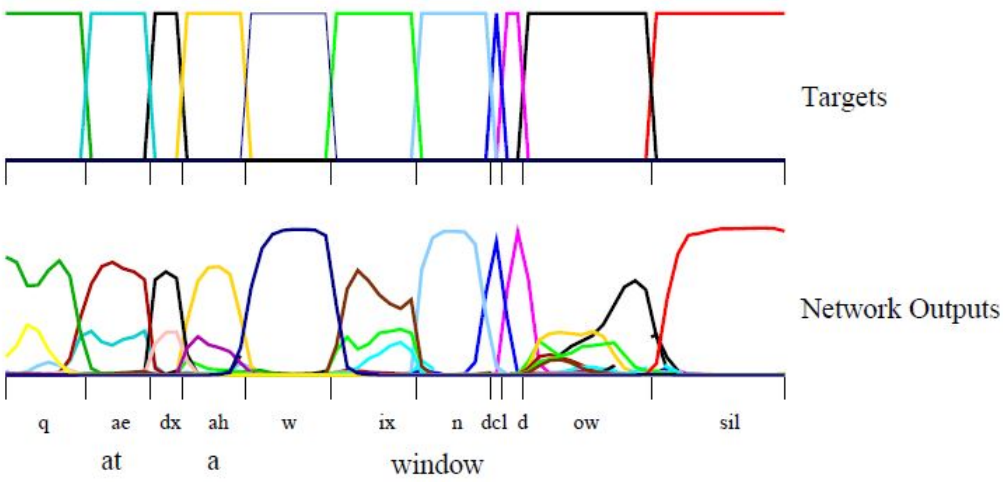


FIGURE 40 – Probabilité pour chaque classe associée à un phonème [21]

Ce graphe représente l'évolution dans le temps des différentes probabilités d'observer un phonème dans le temps. On voit des pics de ces probabilités dans les sorties du réseau de neurones lorsque le phonème est écouté. Le son est donc bien décomposé, les phonèmes sont reconnus et finalement le texte prononcé à l'oral est reconnu.

En supposant, l'indépendance des probabilités y_k^t conditionnellement à l'input x, on obtient alors la distribution suivante pour le 'chemin' $\pi \in L^T$:

$$p(\pi|x) = \Pi_{t=1}^{T} y_{\pi_t}^t$$

avec π étant la séquence constituée en choisissant la classe k à l'instant t. De manière simple, on peut choisir à chaque instant t la classe k ayant la meilleur probabilité.

Considérons l une séquence de lettres en output, connue dans la phase d'entraînement, cette séquence est à faire matcher avec un (ou plusieurs) chemin pour établir un score, nécessaire à l'entraînement du modèle. En effet, un chemin est à différencier d'une séquence de lettres puisqu'il contient encore des éléments non contenus dans la séquence de lettres (*blank* par exemple, ou des doublons de lettres) et qu'il est de manière générale, de taille supérieure à la séquence de lettres.

C'est pourquoi on définit une fonction $\mathbf{F}$ qui permet de relier une séquence l de lettre à un chemin π quelconque. Cette fonction relie l'ensemble des chemins possibles à l'ensemble des séquences possibles. On a, par exemple, $\mathbf{F}(a\_\_ab\_) = \mathbf{F}(\_aa\_\_aab\_\_) = aab$. Puisque tous les chemins sont mutuellement exclusifs, c'est à dire qu'un chemin π est transformé en au maximum une seule séquence de lettres, on peut définir la probabilité

$$p(l|x) = \sum_{\pi \in \mathbf{F}^{-1}(l)} p(\pi|x)$$

qui est la probabilité de reconnaître la séquence de lettre l sachant l'input x.

On peut alors entraîner le modèle avec comme fonction objectif la maximisation de la log vraisemblance du modèle c'est à dire la maximisation de la log probabilité de $p(l|x)$, ce sera notre score. Le calcul de cette probabilité est difficile du fait que la somme sur tous les chemins, est très complexe d'un point de vue algorithmique. Un algorithme de type forward backward similaire à celui utilisé dans les HMMs (voir 7.2) est alors adapté à notre problème, ce qui rend le calcul plus aisé. Le calcul du gradient de cette fonction se fait de la même manière ce qui permet au modèle d'être entraîné par une descente de gradient sur le problème de maximisation de la log-vraisemblance.

L'étape de regroupement des différents chemins dans la même séquence de lettres (pour le calcul de la probabilité $p(l|x)$) est ce qui permet aux réseaux d'utiliser des données non segmentées. En effet, au lieu d'aligner une séquence de lettres en face d'une séquence en input (ce qui impliquerait qu'ils ont la même longueur, au multiple d'une constante près), ce sont les chemins (menant à la même séquence de lettre) qui sont alignés en face de l'input (voir Fig.41).

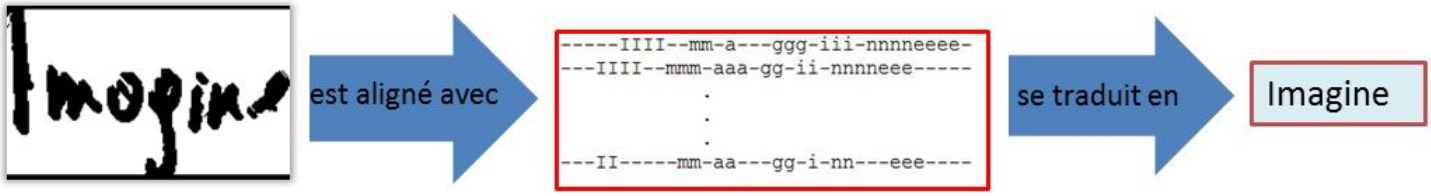


FIGURE 41 – Alignement de l'input et de l'output pour l'entraînement du réseau

3.6.5 Structure d'un réseau de neurones adaptée à la reconnaissance d'écriture

Graves propose un réseau s'appelant *Multidimensional recurrent neural networks MDRNN* ([14]) ayant la particularité de modéliser des connexions multidimensionnelles qui sont exis-

tantes dans les images de texte. Ce réseau est constitué de couches récurrentes multidimensionelles dans les deux directions (backward et forward) qui vont transformer les images 2D en séquence 1D au fur et à mesure de l'avancée dans le réseau puis finalement renvoyer comme output par l'intermédiaire d'une couche de réseau de neurones CTC les probabilités voulues et le mot le plus probable retrouvé.

Dans les expérimentations qui ont été faites de ces réseaux, que ce soit dans la reconnaissance vocale, ou dans la reconnaissance d'écriture manuscrite, les résultats obtenus sont les meilleurs parmi tous les modèles proposés [14].

Dans son article [27], Pham décrit un réseau de neurones capables de faire de la reconnaissance manuscrite. Il utilise les mêmes techniques qu'Alex Graves en ajoutant une parallélisation de certaines couches de neurones (voir Fig.42)

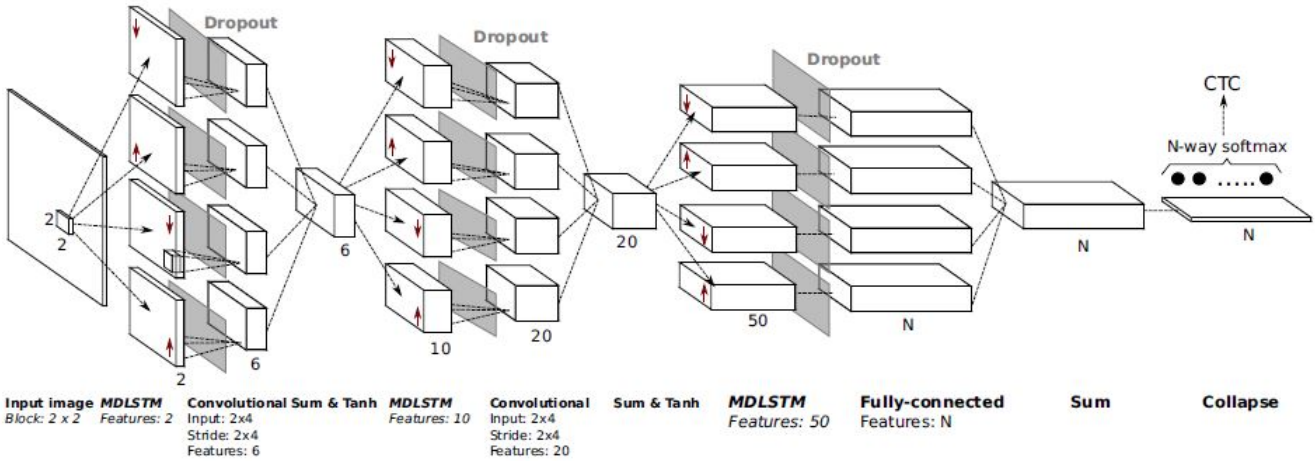


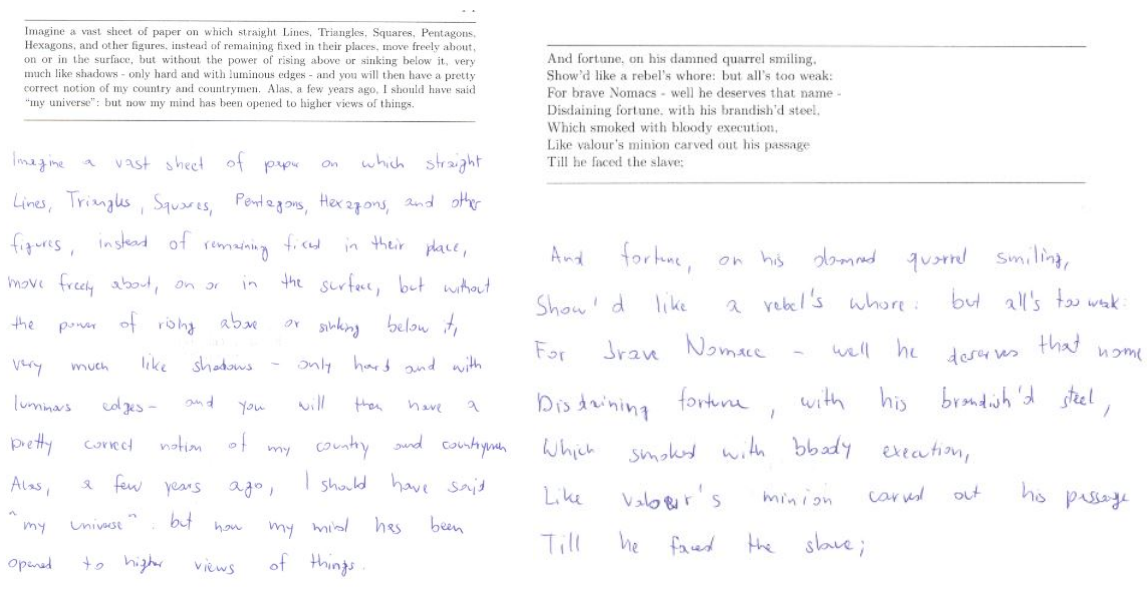
FIGURE 42 – Réseau de neurones proposés dans [27]

Vu Pham et al. vont essayer différentes techniques utilisées dans les réseaux de neurones et notamment le *dropout* pour essayer de voir si elles fonctionnent également dans le cadre d'un problème d'écriture manuscrite. Le dropout est une technique qui permet d'éviter l'overfitting en mettant de coté de manière temporaire certains neurones lors de l'entraînement. Les résultats de ce réseau seront comparés aux nôtres.

3.7 Expérience et résultats

3.7.1 Base de données

Dans le cadre de l'implémentation des modèles proposés à de fins de reconnaissance d'écritures manuscrites, nous avons été amené à utiliser de larges bases de données contenant des images de mots manuscrits. Nous avons ainsi utilisé la base CVL [18] et la base [15].
La base CVL est une base publique destinée à des taches de détection d'écriture, repérage de mots et reconnaissance de manière d'écrire. La base est constituée de 7 textes (6 en anglais et 1 en allemand) écrit par 310 personnes, 27 personnes ont écrit 7 textes et 283 personnes ont écrit seulement 5 textes, tout cela pour un total de 99 904 mots écrits et labelisés. Pour chaque texte, on a une image de couleur RGB (300 dpi) (voir Fig.43)



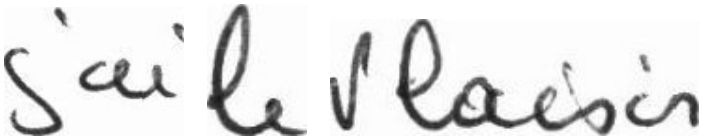
(a) Edwin A. Abbot – Flatland : A Romance of Many Dimension (92 words)

(b) William Shakespeare – Mac Beth (49 words)

FIGURE 43 – Exemple de documents utilisés avec de l'écriture manuscrite, le label étant dans la partie haute de l'image, Database CVL

Le texte initial est placé entre deux séparateurs horizontaux et les personnes ont retranscrit le texte sur la partie basse de l'image en utilisant une aide pour faire des lignes horizontales. La mise en page ressemble à celle de la database IAM [23] aussi utilisée dans le cadre de la reconnaissance manuscrite.

La deuxième database utilisée est la databese RIMES pour Reconnaissance et Indexation de données Manuscrites et de fac similÉS. Elle a été créée pour évaluer les systèmes automatiques de reconnaissance de lettres manuscrites, par exemple les courriers ou fax envoyés par des particuliers à des entreprises ou des administrations. La base de données a été construite en demandant à des volontaires de simuler une demande parmi les demandes suivantes : changement d'informations personnelles, demande d'informations, ouverture ou fermeture de comptes, modification d'un contrat ou d'une commande, plaintes, problèmes de paiement, lettres de rappel, déclarations de dommages. Les volontaires ont la possibilité de choisir les mots qu'ils souhaitent. La collecte des données fut un succès avec plus de 1300 personnes écrivant plus de 5 scénarios pour un total de 12 723 pages. Cette base de données a été utilisée pour différents *data challenge* ICFHR 2008, ICDAR 2009 (reconnaissance de mots), ICDAR 2011 (reconnaissance de lignes).



(a) "j'ai" (b) "le" (c) "plaisir"

FIGURE 44 – Données provenant de la database RIMES

La différence majeur entre ces deux bases se trouve dans le nombre de mots différents présents dans la base. CVL en comporte à peu prés 300 et plus de 5000 pour RIMES.

3.7.2 Choix du réseau de neurones

Le réseau de neurones que nous allons entraîner contient des couches de convolutions et des couches récurrentes. C'est un réseau proposé par le site de documentation d'un package python faisant du Deep-Learning [29]. Ce réseau n'a cependant jamais été utilisé sur de vrais images, les résultats présentés sur le site proviennent d'images simulés très similaires à de l'écriture tapuscrite. Dans cette partie, nous allons utiliser le réseau sur de vraies images d'écriture manuscrite.

Le choix de ce type d'architecture est explicable ainsi. Les couches de convolutions vont permettre d'extraire des *features* (variables explicatives) à partir des images dans un premier temps puis les couches récurrentes vont modéliser l'aspect séquentiel présent dans les features de l'image. Finalement, le réseau utilise un CTC layer comme output layer. Contrairement au MDRNN d'Alex Graves [14], l'aspect séquentiel du réseau ne sera pris en compte qu'après l'extraction de features par les couches de convolutions. La figure 45 montre l'architecture du réseau de neurones choisis pour effectuer la reconnaissance d'écriture manuscrite.

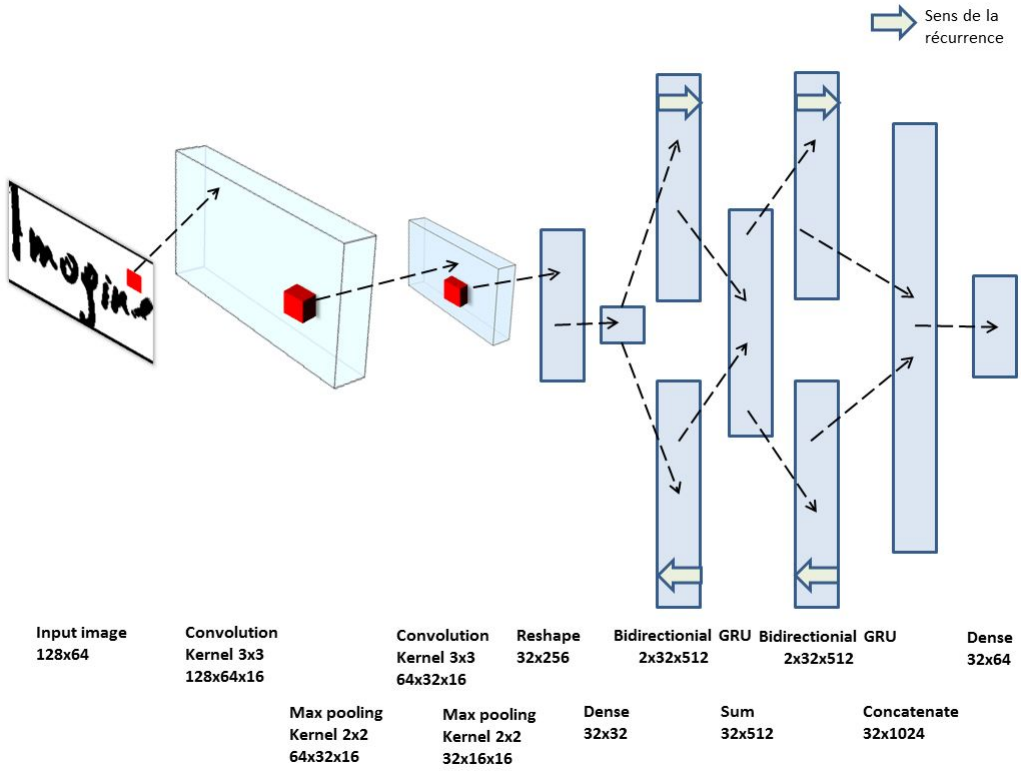


FIGURE 45 – Réseau de neurones construit et implémenté

Le réseau débute par deux couches de convolution chacune suivie d'un max pooling. Le max pooling n'est pas représenté sur la figure 45. Le stride choisi dans la convolution est de 1 et de 2 dans le max pooling, ce qui explique la division par 4 de la dimension de l'input après les deux couches de convolutions. La première couche de convolution applique 16 filtres de convolutions ce qui engendre la dimension 16 de la matrice de sortie des convolutions.

A la suite du deuxième max pooling, l'output est redimensionné de façon à être une matrice à deux dimensions puis un couche dense va réduire la dimension de cette matrice dans le sens qui n'est pas le sens temporel. On note que la dimension temporelle est préservée depuis le début du réseau (elle a juste été réduite). S'ensuit une couche de neurones récurrente GRU bidirectionelle avec somme des deux matrices de sorties et une deuxième couche GRU bidirectionelle

29. `https://github.com/fchollet/keras/blob/master/examples/image_ocr.py`

avec concaténation des matrice de sortie cette-fois ci. Nous décidons de l'utilisation de 512×30 neurones de type GRU pour chaque direction de la séquence. Le réseau finit avec la couche de sortie CTC que nous avons détaillé plus haut. Cette couche n'est rien d'autre qu'une couche dense adaptée à notre problème.

Les paramètres du modèle à optimiser sont à peu prés de 4.9 millions. Le nombre de paramètres à optimiser par couches est détaillé en annexe (voir Annexe 7.4).

3.7.3 Format de l'image en entrée du réseau de neurones

L'image fournie en entré du réseau de neurones est de taille 128×64. De plus, chaque image est rendu binaire avant d'être fourni au réseau. Les images des bases de données ayant une taille supérieur sont rédimensionnées, celles qui ont une taille inférieur sont remplis de 0 sur les bords afin d'obtenir une image de taille 128×64. Dans la dimension de la largeur (64), il n'est pas obligatoire, d'avoir une image ayant une largeur exactement égale à 64. En effet, l'aspect séquentiel du réseau et la couche de neurones de sortie CTC permettent d'avoir des images de largeurs inférieurs à 64.

3.7.4 Choix des méta-paramètres

Deux méta-paramètres interviennent dans la structure du réseau : le nombre de convolutions qui est de 16 et le nombre de séquence de neurones GRU qui est de 512. On peut les choisir différemment mais ces paramètres nous ont donné les meilleurs résultats.

Afin d'entraîner correctement le réseau, certains meta-paramètres sont à choisir judicieusement. Le nombre d'images utilisées pour chaque itération de la back-propagation (*minibatch_ size*) est de 32, c'est assez élevé pour mettre à jour l'ensemble des paramètres du réseau en faisant la moyenne des gradients et ne pas prendre trop en compte d'éventuels outlier (valeurs aberrantes) sans être trop élevé pour noyer le signal. Le learning_rate choisi est de 0.01 et va diminuer de $10^{-}6$ au fur et à mesure de l'entraînement.

Le nombre de fois où l'ensemble de la base de données est utilisé pour entraîner le modèle (*epoch_ size*) sera choisi en utilisant l'early_stopping, cela veut dire que si les résultats se dégradent, au bout d'un certain temps, l'entraînement est arrêté et le meilleur modèle conservé est choisi.

L'ensemble des caractères que nous reconnaissons sont les caractères suivants :
ABCDEFGHIJKLMNOPQRSTVWXYZabcdefghijklmnopqrstvwxyz0123456789 ce qui fait 62 classes ainsi qu'une classe contenant toutes les lettres ayant un accent ou les caractères qui ne sont pas de l'alphabet latin tels que "/" ou "-". La dernière classe du réseau correspond à la classe blanc de l'output layer CTC.

3.7.5 Utilisation de GPU pour l'entraînement du réseau

De manière générale, un ordinateur est constitué d'un processeur (CPU) et d'une carte graphique (GPU), les calculs étant faits par le CPU et le traitement graphique étant fait par le GPU. Dans la cas de réseaux de neurones, il a été démontré que la capacité de calcul de GPU est supérieur à celle des CPU (voir Annexe 7.5). Un réseau de neurones s'entraîne grâce à des millions de calculs simples qui sont pratiqués plus facilement par une carte graphique qu'un processeur.

A des fins de rapidité, l'entraînement du réseau présenté plus haut s'est fait sur une instance virtuelle (ordinateurs virtuels) d'*Amazon Web Service* (AWS). De plus, Amazon a récemment mis en place des instances déjà configurée pour réaliser du Deep-Learning [30]. Les package python pour le Deep-Learning très courant étant tous déjà présent dans l'instance, nous utiliserons *keras* [31] à des fins de simplicité du code. Le choix du type de l'instance est *g2.2xlarge*. Ce type d'instance contient une carte graphique, 8 CPU avec une mémoire vive de 15 GiB et de 60 GB de mémoire. Les CPU sont des processeurs Intel Xeon E5-2670 (Sandy Bridge) [32] et le GPU est une carte graphique NVIDIA avec 1,536 *CUDA* cores et 4GB de mémoire vidéo. Les CUDA *Compute Unified Device Architecture* core sont des technologies développé par NVIDIA qui accélère le processus de calcul du GPU.

3.7.6 Mesure de précision

Afin de mesurer la qualité des modèles, on définit deux mesures de précision : le *Character Error Rate (CER)* et le *Word Error Rate (WER)*. Le CER est calculé en divisant la distance d'édition entre le vrai mot présent dans l'image et le mot reconnu, par la longueur du vrai mot.

$$CER = \frac{1}{\#(prediction, target)} \sum_{(prediction, target)} \frac{edit\_distance(prediction, target)}{|target|}$$

La distance d'édition entre deux mots consiste en le nombre d'insertions, de suppression et de remplacements de lettres à faire, pour faire matcher les deux mots. Le WER est simplement le taux d'erreur de classification des mots.

$$WER = \frac{\#(prediction \neq target)}{\#prediction}$$

3.7.7 Résultats

Dans son livre [12], Graves expose les résultats obtenus en utilisant des réseaux de neurones et ceux obtenus avec les HMM. Les réseaux de neurones surperforment les HMM dans l'ensemble des tests et sur l'ensemble des bases d'entraînement.

Nous présentons dans la table 2 les résultats que nous avons obtenus en entraînant nos propres réseaux. Nous décidons d'un découpage train/validation de 90%/10%. La base de validation est utilisée pour étudier l'amélioration des résultats au cours de l'entraînement. La base de test sera dans le cas de l'entraînement sur Rimes et dans le cas de l'entraînement sur Rimes+CVL, la base de test fournie par Rimes tandis que dans le cas de l'entraînement sur uniquement CVL, ce sera la base de validation de CVL qui sera également base de test.

A chaque fois que nous entraînons un modèle, nous ajoutons une centaine d'image blanche pour l'entraînement. Cela permet d'aider le réseau à détecter les zones sans écriture des vraies images. Les résultats de la table 2 sont obtenus en entraînant le modèle sur la base train et en évaluant sur la base test.

30. `https://aws.amazon.com/fr/amazon-ai/amis/`
31. `https://keras.io/`
32. `http://www.bit-tech.net/reviews/tech/cpus/intel-xeon-e5-2670-review/1/`

Article	Rimes CER	Rimes WER	CVL CER	CVL WER	Rimes+CVL CER	Rimes+CVL WER
LSTM 30 [27]	14.72	42.03	-	-	-	-
LSTM 50 [27]	15.11	42.62	-	-	-	-
LSTM 100 [27]	15.79	44.37	-	-	-	-
LSTM 200 [27]	**14.68**	42.07	-	-	-	-
Our neural network	25.0	48.5	12	28	29.2	54.3
Our neural network with correction	17.11	**36.3**	-	-	-	-

TABLE 2 – Résultats du réseau de neurones

Les résultats de la reconnaissance de l'écriture manuscrite avec le modèle entraîné sur CVL, sur des images provenant de la base de validation issue toujours de la base CVL sont excellents (12 en CER et 28 en WER). Cependant le modèle n'est pas efficace sur des mots qui ne sont pas dans la base de données (ne se voit pas sur le tableau de résultats). Ceci s'explique par le fait que la base CVL comprenant uniquement 290 mots différents, le réseau en vient à faire presque une tache de classification sur les mots. Ce biais était prévisible puisque la base CVL n'est à l'origine pas adéquate pour une tache de reconnaissance d'écriture manuscrite. Elle est surtout utilisée pour reconnaître un type ou un style d'écriture.

Les résultats obtenus sur la base Rimes sont satisfaisants (25 en CER et 48.5 en WER), un peu plus faibles que ceux obtenus par l'article [27]. Cependant, notre réseau a été entraîné sans les classes d'accents et autre caractères spéciaux ce qui entraîne un nombre important d'erreurs. On note également que le choix d'une base de validation de grande taille pénalise un peu la reconnaissance. En ajoutant un correcteur orthographique, on s'aperçoit que le score WER diminue fortement. Ce correcteur est un correcteur générique n'utilisant pas la matrice des probabilités. Il se base sur un dictionnaire français et corrige les mots non présents dans le dictionnaire en leur associant le mot le plus proche tout en privilégiant les mots fréquemment utilisé dans la langue française. Par sa nature, ce correcteur est adapté uniquement à la langue française. On obtient ainsi 39% de mots mal reconnus en utilisant le correcteur. En affinant les résultats, on peut voir que même 70% des mots n'ont au maximum qu'une seul erreur. Voici un graphique affinant encore cette analyse :

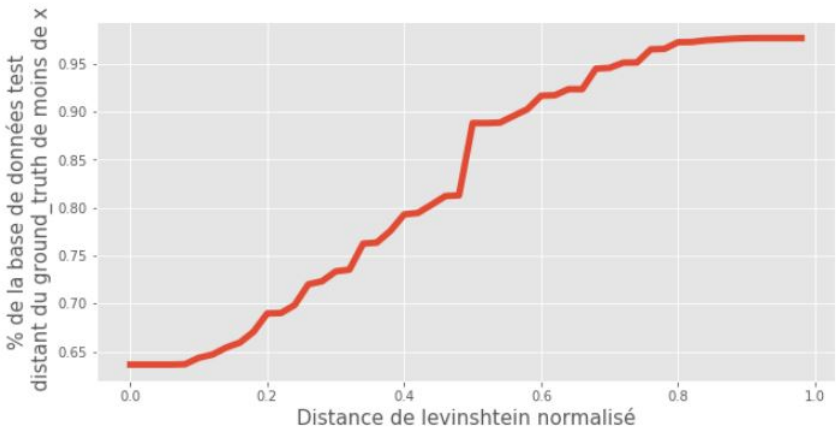


FIGURE 46 – % de mots ayant une distance de levenshtein normalisé vis-à-vis du vrai mot inférieur à x

On a donc près de 64% des mots qui sont totalement reconnus, 70% des mots qui sont reconnus à raison de 8 lettres sur 10 bien reconnus.

En concaténant les deux bases, Rimes et CVL, on obtient des résultats plus mauvais dans la reconnaissance sur Rimes. La diversité de la base Rimes et la taille de la base CVL nous laissaient penser que cela engendrerait une base de données plus complètes pour l'entraînement du réseau. Au contraire le biais de la base CVL est à l'origine de la détérioration des résultats. De plus, CVL étant en anglais et Rimes en français, cela peut engendrer une incompréhension du réseau dans la recherche de l'aspect systématique de la séquence d'un mot.

Par rapport au temps de convergence, une dizaine d'epoch suffisent seulement pour entraîner correctement le réseau. La figure 48 montre l'évolution des scores au cours de l'entraînement du réseau.

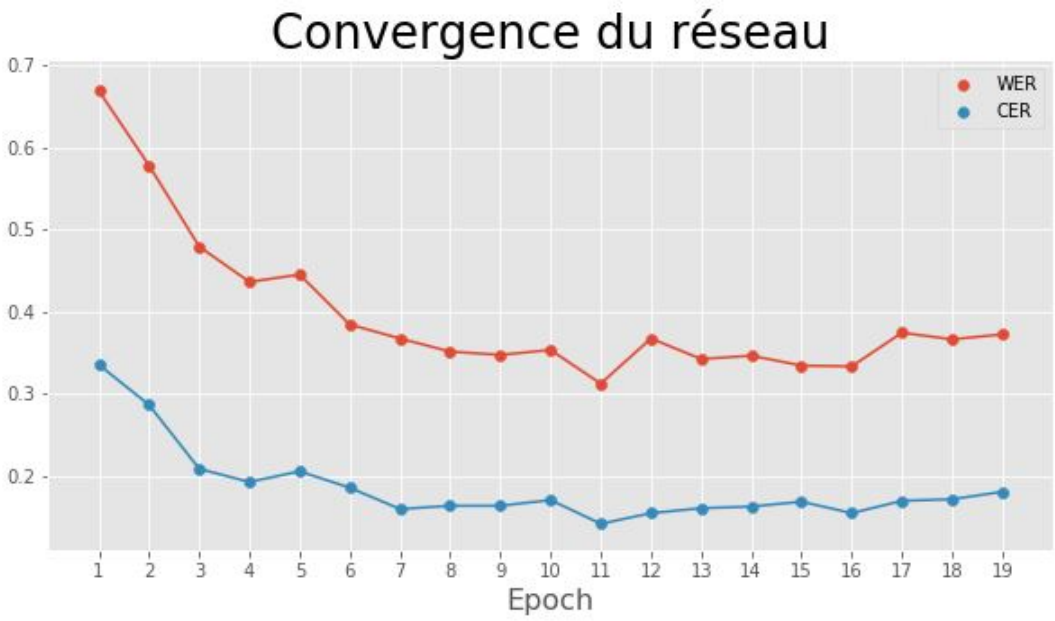


FIGURE 47 – Convergence des valeurs de WER et CER au cours de l'entraînement

On peut considérer que les réseaux de neurones traitent bien le problème de la reconnaissance.

Pour avoir un aperçu plus clair des résultats et des erreurs que font les réseaux, voici un exemple de résultats obtenus sur une phrase issue de la base de données de test de Rimes.

FIGURE 48 – Résultats de la reconnaissance de l'écriture manuscrite par les réseaux de neurones

Le réseau utilisé ici est celui qui a été seulement entraîné sur la base Rimes. Les mots courts ont la plupart été bien reconnus, les mots longs ont une ou deux lettres qui n'ont pas été reconnues ou détectées.

3.7.8 Limites et améliorations du modèle

Les erreurs visibles ici sur la figure 48 sont celles de caractères spéciaux qui ne sont pas reconnues dans les réseaux. Cela est prévisible puisqu'il n'y a pas de classe spécifique à ces caractères, seul une classe 'else' reconnaît la présence d'un caractère inconnu. Notons que le mot "else" impacte fortement le CER en l'augmentant. On peut imaginer mettre un espace blanc chaque fois que la classe 'else' est reconnu. De même, les lettres avec accents ne sont pas une classe particulière dans le réseau, elles ont été identifiées dans la base d'entraînement comme appartenant à la classe des caractères inconnus. On peut améliorer le réseau en les ajoutant comme classes particulières. Toutefois le réseau n'est alors plus exactement adapté à la langue anglaise.

Le 14 éme mot de la figure 48 a été mal reconnu à cause du fait que l'espace entre les deux lettres 'l' n'a pas été détecté, ce qui implique que la séquence de classes les plus probables obtenue par le réseau était du type 'eeee—-llllllll–eeess' ce qui se traduit par 'eles'. Pour obtenir, le bon résultat, la séquence devrait être du type 'eeee—-llll—llll–eeess'. On rappelle que la séquence des classes est obtenue en prenant le meilleur chemin sur la matrice des probabilités de type 39, voir la partie 3.6.4.

Afin d'augmenter le nombre de mots bien reconnus, c'est à dire le WER, le réseau de neurones doit pouvoir utiliser un dictionnaire se basant sur les probabilités issues du réseau de neurones pour vérifier l'orthographe des mots qu'il reconnaît. Ce correcteur orthographique n'est plus le simple correcteur que nous utilisé précédemment. Ce correcteur permettrait au réseau de sélectionner le mot le plus probable mais qui est également présent dans le dictionnaire. Ainsi, au lieu d'utiliser la méthode du meilleur chemin dans l'analyse des probabilités fournies par le réseau, on étudierait le meilleur chemin associé à un mot du dictionnaire français ou anglais.

4 Exemple d'applications à l'assurance de la reconnaissance automatique des lettres

4.1 Mutualisation des taux minimum garantis

L'essor des mathématiques appliquées à l'assurance et des sciences actuarielles ainsi qu'une meilleur compréhension des risques pris par les assureurs a entraîné l'élaboration de nouvelles régulations en Europe et dans le monde. Ainsi, certains assureurs ont travaillé sur la création de modèle ALM dynamique et stochastique permettant la prédiction des flux d'argent futurs et l'estimation des provisions techniques.

Dans la conception de son modèle, un assureur A a identifié la nécessité d'analyser dans le détail les règles du partage des profits entre les différents contrats de ces clients. Milliman a donc eu comme mission d'analyser les différents contrats d'assurance (stockés sous formes de scans) et d'estimer les conséquences d'une mutualisation des profits. Afin d'automatiser cette reconnaissance, un outil d'analyse et de reconnaissance textuelle est nécessaire. La configuration du modèle ALM pour prendre en compte cette mutualisation et une quantification des impacts globaux sur le modèle de tels changements ont été les dernières étapes de la mission.

De manière plus précise, auparavant dans le modèle ALM de l'assureur A, sauf cas de taux cible faible, les contrats ne pouvaient pas se mutualiser c'est-à-dire que chaque contrat ne pouvait recevoir moins que la valeur $\max(\text{TMG}, \text{pb\_contra})$ où TMG est le *taux minimum garanti* à l'assuré et pb_contra est la participation aux bénéfices du contrat. Il était donc exclu de pouvoir financer les contrats à TMG élevés à l'aide des contrats à TMG faibles en cas de revenus financiers insuffisants. Par exemple si un contrat X a un TMG de 4% et une pb_contra de 2% et un autre contrat Y a un TMG de 1% et une pb_contra de 3% , si l'on ne mutualise pas, le contrat X va recevoir 4% et le contrat Y va recevoir 3%. Avec la nouvelle règle, le contrat Y devrait recevoir moins de 3% au titre de la participation aux bénéfices afin de financer le TMG du contrat X. On résume cela dans le tableau 3.

	TMG	Pb_contra	Sans mutualisation	Avec mutualisation
Contrat X	4%	2%	4%	4%
Contrat Y	1%	3%	3%	<3%

Table 3 – Exemple d'une mutualisation des TMG

Toutefois les contrats d'assurance n'ont pas tous la possibilité de mutualiser leur TMG. Cela dépend des conditions générales du contrat. Chaque contrat doit donc d'abord être étudié par un opérateur qui identifie les clauses de partage du profit et qui indiquent quels sont les éléments permettant ou pas une mutualisation des TMG.

Milliman a donc étudié l'ensemble des contrats qui étaient des scans ou des contrats papiers "à la main" et les a classé en trois catégories "mutualisation possible" (catégorie 1), "la clause est sujette à interprétations" (catégorie 2) et "mutualisation impossible" (catégorie 3) (voir Fig.49). La masse des contrats étant très importante, seul un échantillon a été étudié.

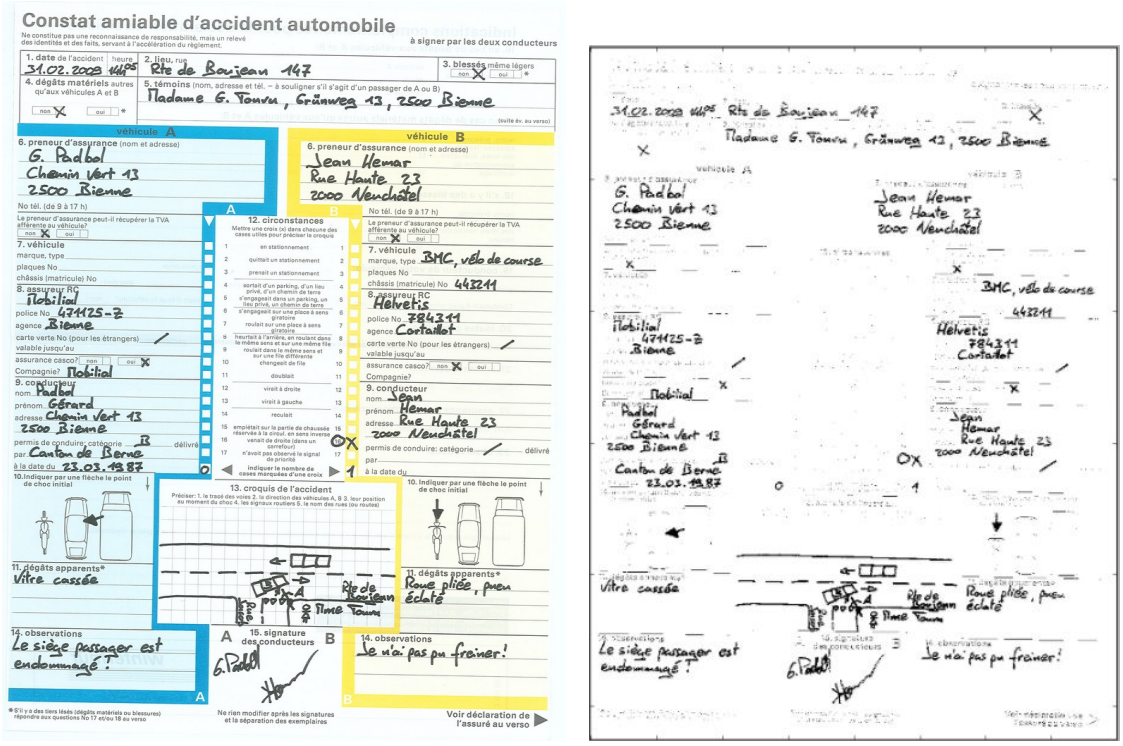
Grâce à la reconnaissance automatique de l'écriture tapuscrite et manuscrite, il aurait été possible dans un premier temps de transformer chaque scan de contrat en un fichier texte puis en utilisant des méthodes de text minning présenté plus tard dans ce mémoire, extraire des mots-clés ou des paragraphes relatifs à la mutualisation.

	Classification des clauses de participation aux bénéfices Détails		
Numero	Libellé de la clause relative à la participation aux bénéfices	Segmentation retenue	% PM
1	Le 31 décembre de chaque année, la participation aux bénéfices est déterminée par l'affectation au contrat d'au moins 90% des résultats nets des placements réalisés par l'actif général de [redacted] Le conseil d'administration de la société détermine les conditions d'affectation des bénéfices techniques et financiers. Le montant ainsi obtenu est diminué des frais de gestion annuels (0,75%) prélevés prorata temporis. Le taux de rendement ainsi attribué ne peut être inférieur au taux de rendement minimum annuel défini au 1er alinéa.	Catégorie 2	1,72%
2	De la participation aux bénéfices, déterminée par l'affectation d'au moins 90% des résultats techniques et des résultats nets des placements réalisés par l'actif général de [redacted] Le conseil d'administration de la société détermine les conditions d'affectation des bénéfices techniques de financiers ainsi que leur attribution à chaque catégorie de contrat et à la provision pour participation aux bénéfices.	Catégorie 1	33,81%
3	Au 31 décembre de chaque année, l'épargne constituée est augmentée de la participation aux bénéfices, déterminée par l'affectation au contrat d'au moins 90% des résultats nets des placements réalisés par l'actif général de [redacted]	Catégorie 2	2,23%
4	Au 31 décembre de chaque année, l'épargne constituée est augmentée de la participation aux bénéfices, déterminée par l'affectation au contrat de 100% des résultats nets des placements réalisés par l'actif général de la Compagnie.	Catégorie 2	0,03%
5	Le 31 décembre de chaque année, une participation aux bénéfices est attribuée. Elle est déterminée par l'affectation au contrat d'au moins 90% des résultats nets des titres venant en représentation de [redacted] Ces titres font l'objet d'une compatibilité divisionnaire. Le montant ainsi obtenu est diminué des frais de gestion.	Catégorie 3	0,13%

FIGURE 49 – Exemple de clauses de partage de profits

4.2 Constat d'accident routier

L'une des applications possibles pour la reconnaissance d'écriture manuscrite est la lecture automatique des constats. Les assureurs automobiles reçoivent tous les jours des centaines de constats automobiles dues à la collision entre deux véhicules ou entre un véhicule et une personne. Les conducteurs des véhicules établissent le constat en écrivant à la main l'ensemble de leurs coordonnées et une description succincte de la situation qui a engendré la collision. Voici l'exemple d'un constat français trouvé sur internet :



(a) Constat automobile

(b) Constat où les pixels ont été seuillés

FIGURE 50 – Constat

Pour extraire automatiquement l'écriture manuscrite, on pourrait superposer cette image à l'image d'un constat vide et utiliser la différence pour n'avoir que les écritures manuscrites. Cela demande une attention particulière sur la qualité du scan car si les qualités du constat vierge et du constat rempli diffère, la superposition est beaucoup plus compliquée. On peut également appliquer un filtre pour récupérer uniquement les pixels provenant de l'écriture (voir Fig.50).

On peut ensuite chercher les zones de texte puis les lignes et enfin les mots et les reconnaître les uns après les autres en suivant le pipeline défini pour l'écriture manuscrite. Le réseau utilisé ici est celui entraîné sur Rimes. Les images des mots ont été travaillées de manière à diminuer l'épaisseur de l'écriture.

Toutefois cela n'a pas été suffisant pour obtenir de bons résultats (voir Fig.51), seules certaines syllabes de certains mots ont été reconnues. Une raison à cela est que les images des mots issus du constat sont vraiment très différentes de celle de la base d'entraînement de Rimes. L'épaisseur de l'écriture, (malgré le traitement de l'image) et le bruit sur les images redent impossible leur reconnaissance par le réseau de neurones. En effet, les réseaux de neurones sont des modèles assez sensibles aux bruits si leur entraînement n'est pas fait sur une base de données comprenant des donnés bruitées.

Afin d'obtenir de meilleurs résultats, il faudrait comme pour le texte tapuscrit, avoir une base d'entraînement beaucoup plus grande, regroupant différentes types d'écriture avec différentes épaisseurs et différents styles d'écriture. Ce type de base de données n'existe pas en open-source. La base Rimes ne possédait que 57000 exemples, ce qui est relativement faible. Pour donner une idée du nombre de données qu'il faudrait pour entraîner le réseau à la perfection, Baidu, le concurrent de Google chinois, a utilisé près de 13000 heures de données audio, ce qui représente des millions de données labellés pour entraîner un réseau de neurones qui fait du speech-to-text

de manière assez similaire au nôtre.

FIGURE 51 – Résultat sur mots extraits du constat

4.3 Acte de naissance et certificat de décès

Les certificats décès établis dans les années 60 sont pour la plupart des questionnaires complétés à la main par un médecin. Ces certificats contiennent les donnés personnelles des personnes décédées ainsi qu'une explication de la cause de leur décès. Pour comprendre la mortalité des années d'après guerre et faire une analyse statistique des causes de décès, il est nécessaire d'avoir accès à une large base de certificats de décès, par exemple certains certificats de décès sont librement accessibles en lignes [33] (voir Fig.52). La constructions de tables de mortalité ne peut être envisagée sur l'étude de ces simples certificats du fait de la censure et de la troncature des données mais aussi à cause du biais, les certificats accessibles en ligne peuvent ne représenter qu'une catégorie de personnes. Toutefois, une compréhension de la mortalité des années 60 peut se faire sur la base simple d'une étude statistique.

33. http://genealogy.az.gov/

Automatiser la reconnaissance de la cause du décès devient nécessaire dans le cadre de cette étude.

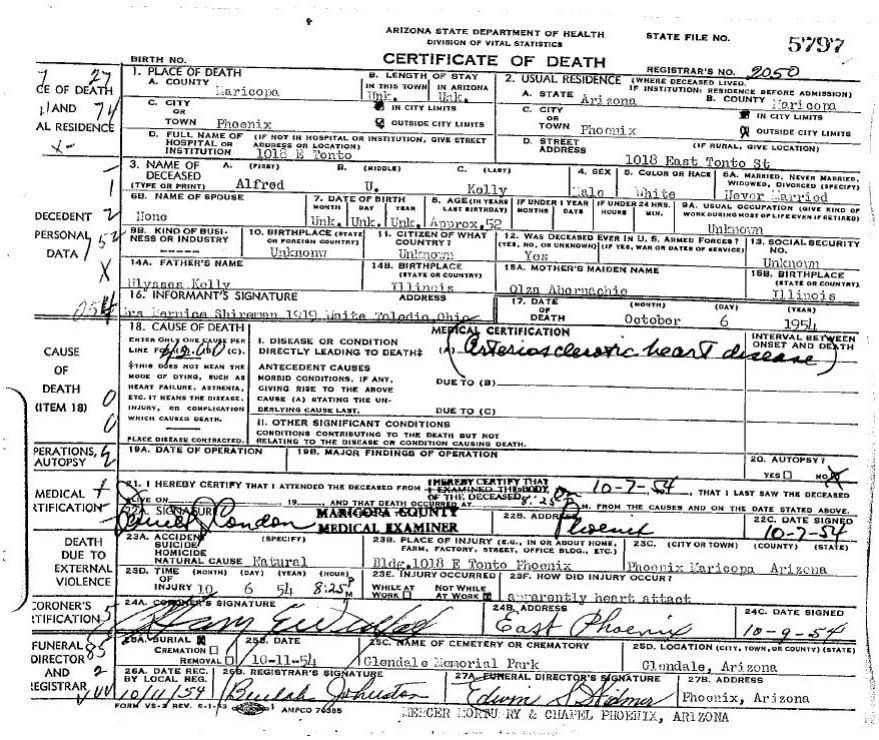

ARIZONA STATE DEPARTMENT OF HEALTH
DIVISION OF VITAL STATISTICS
STATE FILE NO. 5797
CERTIFICATE OF DEATH
REGISTRAR'S NO. 2050
BIRTH NO.
1. PLACE OF DEATH A. COUNTY Maricopa B. LENGTH OF STAY IN THIS TOWN Unk. IN ARIZONA Unk.
C. CITY OR TOWN Phoenix IN CITY LIMITS OUTSIDE CITY LIMITS
D. FULL NAME OF HOSPITAL OR INSTITUTION (IF NOT IN HOSPITAL OR INSTITUTION, GIVE STREET ADDRESS OR LOCATION) 1018 E Tonto
2. USUAL RESIDENCE (WHERE DECEASED LIVED. IF INSTITUTION: RESIDENCE BEFORE ADMISSION) A. STATE Arizona B. COUNTY Maricopa
C. CITY OR TOWN Phoenix IN CITY LIMITS OUTSIDE CITY LIMITS
D. STREET ADDRESS (IF RURAL, GIVE LOCATION) 1018 East Tonto St
3. NAME OF DECEASED (TYPE OR PRINT) A. (FIRST) Alfred B. (MIDDLE) U. C. (LAST) Kelly
4. SEX Male 5. COLOR OR RACE White 6A. MARRIED, NEVER MARRIED, WIDOWED, DIVORCED (SPECIFY) Never Married
6B. NAME OF SPOUSE None 7. DATE OF BIRTH MONTH Unk. DAY Unk. YEAR Unk. 8. AGE (IN YEARS LAST BIRTHDAY) Approx. 52 IF UNDER 1 YEAR MONTHS DATE IF UNDER 24 HRS. HOURS MIN.
9A. USUAL OCCUPATION (GIVE KIND OF WORK DURING MOST OF LIFE EVEN IF RETIRED) Unknown
9B. KIND OF BUSINESS OR INDUSTRY ------ 10. BIRTHPLACE (STATE OR FOREIGN COUNTRY) Unknown 11. CITIZEN OF WHAT COUNTRY? Unknown 12. WAS DECEASED EVER IN U. S. ARMED FORCES? (YES, NO, OR UNKNOWN) Yes (IF YES, WAR OR DATES OF SERVICE)
13. SOCIAL SECURITY NO. Unknown
14A. FATHER'S NAME Ulysses Kelly 14B. BIRTHPLACE (STATE OR COUNTRY) Illinois 15A. MOTHER'S MAIDEN NAME Olga Abernachie 15B. BIRTHPLACE (STATE OR COUNTRY) Illinois
16. INFORMANT'S SIGNATURE Ira Bernice Shipman 1919 White Toledo, Ohio ADDRESS
17. DATE OF DEATH (MONTH) October (DAY) 6 (YEAR) 1954
18. CAUSE OF DEATH ENTER ONLY ONE CAUSE PER LINE FOR (a), (b), (c). THIS DOES NOT MEAN THE MODE OF DYING, SUCH AS HEART FAILURE, ASTHENIA, ETC. IT MEANS THE DISEASE, INJURY, OR COMPLICATION WHICH CAUSED DEATH. PLACE DISEASE CONTRACTED.
MEDICAL CERTIFICATION
I. DISEASE OR CONDITION DIRECTLY LEADING TO DEATH* (A) Arteriosclerotic heart disease INTERVAL BETWEEN ONSET AND DEATH
ANTECEDENT CAUSES MORBID CONDITIONS, IF ANY, GIVING RISE TO THE ABOVE CAUSE (A) STATING THE UNDERLYING CAUSE LAST. DUE TO (B) DUE TO (C)
II. OTHER SIGNIFICANT CONDITIONS CONDITIONS CONTRIBUTING TO THE DEATH BUT NOT RELATING TO THE DISEASE OR CONDITION CAUSING DEATH.
19A. DATE OF OPERATION 19B. MAJOR FINDINGS OF OPERATION 20. AUTOPSY? YES NO
21. I HEREBY CERTIFY THAT I ATTENDED THE DECEASED FROM 10-7-54, THAT I LAST SAW THE DECEASED ALIVE ON ..., 19..., AND THAT DEATH OCCURRED AT 8:25 P.M. FROM THE CAUSES AND ON THE DATE STATED ABOVE.
22A. SIGNATURE MARICOPA COUNTY MEDICAL EXAMINER 22B. ADDRESS Phoenix 22C. DATE SIGNED 10-7-54
23A. ACCIDENT SUICIDE HOMICIDE NATURAL CAUSE (SPECIFY) Natural 23B. PLACE OF INJURY (E.G., IN OR ABOUT HOME, FARM, FACTORY, STREET, OFFICE BLDG., ETC.) Bldg. 1018 E Tonto Phoenix 23C. (CITY OR TOWN) Phoenix (COUNTY) Maricopa (STATE) Arizona
23D. TIME OF INJURY (MONTH) 10 (DAY) 6 (YEAR) 54 (HOUR) 8:25 P.M. 23E. INJURY OCCURRED WHILE AT WORK NOT WHILE AT WORK 23F. HOW DID INJURY OCCUR? apparently heart attack
24A. CORONER'S SIGNATURE 24B. ADDRESS East Phoenix 24C. DATE SIGNED 10-9-54
25A. BURIAL CREMATION REMOVAL 25B. DATE 10-11-54 25C. NAME OF CEMETERY OR CREMATORY Glendale Memorial Park 25D. LOCATION (CITY, TOWN, OR COUNTY) (STATE) Glendale, Arizona
26A. DATE REC. BY LOCAL REG. 10/11/54 26B. REGISTRAR'S SIGNATURE 27A. FUNERAL DIRECTOR'S SIGNATURE 27B. ADDRESS Phoenix, Arizona
FORM VS-2 REV. 6-1-53 AMPCO 76595
BERGER MORTUARY & CHAPEL PHOENIX, ARIZONA

FIGURE 52 – Certificat de décès de l'Arizona

Ces documents sont excessivement complexes à étudier du fait de la faible qualité du document et du bruit ajouté à l'image. Des étapes de pré-traitement de l'image sont nécessaires pour avoir une chance de réussir à extraire de l'information de qualité. D'une part, la détection de l'écriture manuscrite sur ce type de documents doit être extrêmement précise à cause du bruit présent sur l'image, le bruit étant les nombreux points noirs isolés sur l'image. D'autre part, ce bruit doit être retiré pour ne pas perturber le programme de reconnaissance d'écriture manuscrite. Malgré tous nos efforts pour rendre la reconnaissance possible, les étapes de pré-traitement que nous avons essayé n'ont pas suffit à l'amélioration de la qualité de l'image des certificats de décès et donc à la reconnaissance.

Dans un cadre un peu similaire, l'étude de la dynamique des populations consiste en l'examen de l'évolution de la taille et de l'âge des populations à travers des modèles théoriques et des simulations. Dans le cadre de la construction des tables de mortalité, ces études sont importantes pour en assurer la qualité. L'exposition dans la construction des tables de mortalités doit être prise en compte afin d'assurer un mesure précise des coefficients q_x. Cette exposition peut être estimé par une étude de la répartition des naissances selon les mois de l'année.
Le site geneanet [34] maintient une base de données contenant des répertoires d'actes de naissance. En voici un exemple,

34. `http://www.geneanet.org`

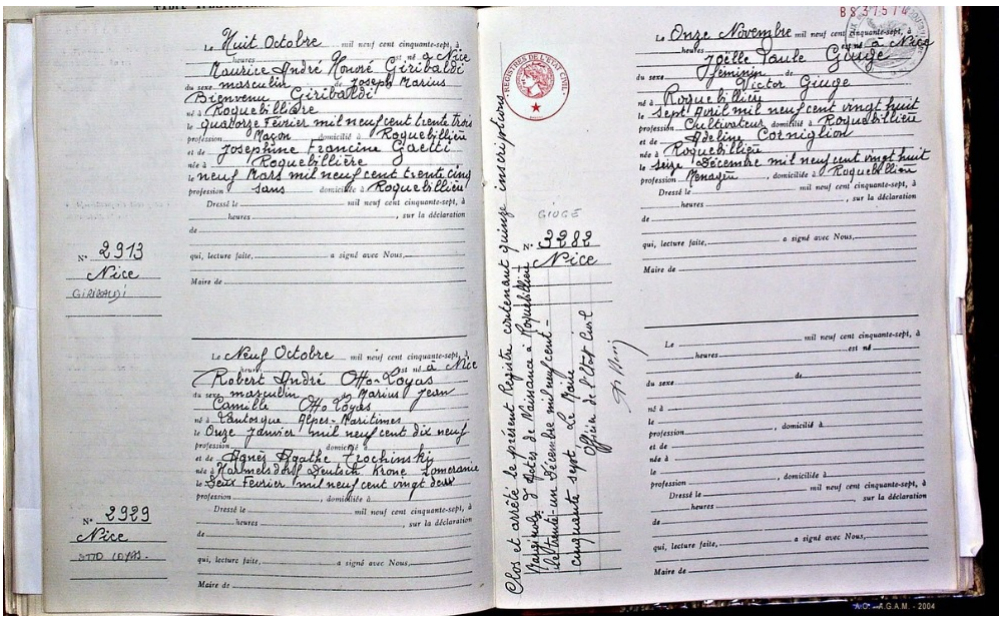


FIGURE 53 – Exemple d'acte de naissance

Certains actes ont été listés à la main par le site mais pour une étude exhaustive de ces répertoires, l'algorithme que nous avons implémenté est indispensable. Les dates se trouvent dans la première ligne de chaque actes. On peut restreindre notre algorithme à la lecture des première lignes et renvoyer la date la plus probable qu'il reconnaît.

4.4 Extraction des informations d'un contrat de réassurance XS of loss

Voici un exemple simple d'une utilisation de l'algorithme de reconnaissance d'écriture tapuscrite. Dans les conditions générales d'un contrat de réassurance, beaucoup d'informations sont fournis. Afin de les extraire de manière consistante, après avoir utiliser l'outil de reconnaissance tapuscrite sur l'ensemble du contrat, on utilise les *regex* sur le texte détecté et reconnu.

En informatique, une expression régulière ou regex est une chaîne de caractères, qui décrit, selon une syntaxe précise, un ensemble de chaînes de caractères possibles. Les expressions régulières sont issues des théories mathématiques des langages formels datant des années 1940. Leur capacité à décrire avec concision des ensembles réguliers explique qu'elles se retrouvent dans plusieurs domaines scientifiques. Les expressions régulières sont aujourd'hui utilisées par les informaticiens dans l'édition et le contrôle de texte ainsi que dans la manipulation des langues formelles que sont les langages informatiques.

Voici par exemple le code python utilisant les regex pour détecter la priorité et la garantie d'un contrat XS of loss (Fig.54).

```
value1="\d+(\d{0,3}\s?)*"
value2="(\d{1,3}\s?)*"
currency="([a-zA-Z]{0,6})"
xs= """
    (?P<garantie>{garantie})
    \s?
    (?P<currency>{currency})?
    \s?
    (XS|X5)
    \s?
    (?P<priorité>{priorité})
    \s?
    (?P<currency>{currency})?
    """
xs=xs.format(garantie=value1,currency=currency,priorité=value2)
xs=re.compile(xs, re.IGNORECASE | re.MULTILINE | re.UNICODE | re.DOTALL | re.VERBOSE)
```

FIGURE 54 – Expression regex et code python pour détecter la priorité et la garantie d'un contrat XS of loss

Ce code permet de définir de quelle type sera la séquence de lettres à détecter. Ici nous spécifions que nous recherchons une séquence de lettre ayant un chiffre suivi d'un mot (pour la devise) puis du mot XS puis d'un nouveau chiffre et enfin un autre mot (pour la devise).

Plan de réassurance non proportionnel à effet du 1er janvier 2015

- ✓ XS tête Tranche 1 : 4 900 000 EUR XS 900 000 EUR
- ✓ XS tête Tranche 2 : 4 200 000 EUR XS 5 800 000 EUR
- ✓ **XS par évènement :**

1ère tranche : **38 800 000 EUR XS 1 200 000 EUR**

2ème tranche : **40 000 000 EUR XS 40 000 000 EUR**

3ème tranche : **40 000 000 EUR XS 80 000 000 EUR**

FIGURE 55 – Définition des conditions techniques d'un contrat XS of loss, document scanné

En utilisant les expressions régulières sur le texte reconnu par Tesseract de l'image présentée en figure 55, on arrive à en extraire les 5 expressions XS avec la garantie et la priorité.

De manière générale dans ce type de contrat, une information peut être répétée plusieurs fois pour envisager des cas différents, c'est le cas des valeurs XS dans la figure 55. Une analyse du contexte dans lequel s'inscrit l'information extraite permet de comprendre l'information et son domaine d'application. Ainsi on peut par exemple chercher à résumer de manière automatique le paragraphe dans lequel se trouve la séquence de mots détectée. Une utilisation du text mining serait utile dans ce cadre. Nous présenterons les méthodologies du text mining et un exemple d'application dans la suite du mémoire.

5 L'analyse de l'information extraite

5.1 Contexte et attente

Dans la première partie du mémoire, nous avons étudié des méthodes et entraîné des modèles à des fins de reconnaissance d'écriture manuscrite ou tapuscrite. Une fois le texte extrait, il s'agit de l'utiliser de manière à en extraire l'information la plus importante. Extraire l'information la plus importante peut se faire de plusieurs manières. Par exemple, les paragraphes importants d'un texte sont détectés puis résumés, ou encore des mots-clés sont extraits automatiquement du texte de manière automatique. Une analyse de type *Natural language processing* (NLP), qui est la manière de travailler sur du texte en utilisant du machine learning, permet cela.

Le texte contrairement à l'image n'est pas une donnée de type numérique. Pour pouvoir appliquer des algorithmes de machine learning sur celui-ci, il faut trouver un espace de représentation numérique de celui-ci. Plusieurs types de représentation existent, certains se basant sur une approche fréquentielle, d'autres sur une approche contextuelle. C'est ce que nous allons présenter dans la partie méthodologique suivante.

Pour des raisons de confidentialité, il est difficile de travailler sur des contrats de tous types et d'en présenter les résultats, par exemple dans le cadre de l'analyse automatique de la signification de certaines clauses.

Par ailleurs, dans la nécessité de gestion des risques par les assureurs, le risque de réputation est l'un des plus difficiles à évaluer. D'une part, parce que la réputation n'est pas quelque chose de quantifiable directement, d'autre part parce que les données pour l'évaluer ne sont pas directement accessibles, il n'existe pas de questionnaires portant sur la réputation d'une entreprise par exemple. Nous proposons donc un moyen d'estimer ce risque, ou au moins d'avoir un indicateur sur les tendances de la réputation des assureurs. Pour illustrer notre étude, nous utiliserons des commentaires d'assurés postés sur le site *opinion-assurance.fr*[35]. Ce site permet aux internautes d'évaluer leurs assureurs et de poster un commentaire sur leur expérience. Ces commentaires nous permettront d'étudier le comportement et les réactions des assurés vis-à-vis de leur compagnie d'assurance et donc d'estimer la réputation de chaque assureurs chez ces personnes. De plus, ces commentaires sont lues des millions de fois, cela implique donc l'évaluation d'un risque de réputation à grande échelle.

5.2 Méthodologie en Text-Mining

Dans cette partie, nous exposerons les méthodes nécessaires au travail sur de la donnée textuelle. Un texte est reconnu par un ordinateur comme une séquence de lettres et de ponctuations appelé *string*. Cette séquence n'est pas utilisable de la sorte par des algorithmes de machine-learning.

Afin de transformer les mots ou les phrases en vecteur numériques, il s'agit de trouver un espace vectorielle dans lequel les mots peuvent être représentés. Cet espace vectoriel devra en plus introduire une notion de distance entre les mots ou les phrases. Cette distance peut se baser sur une approche fréquentielle : si deux mots sont souvent dans le même document, alors ces mots sont proches. Cette distance peut se baser sur une relation logique entre les mots. Ainsi le mot "France" est proche du mot "Angleterre" puisque les deux mots représentent un pays.

35. `https://www.opinion-assurances.fr/`

5.2.1 Preprocessing d'un texte

Avant de chercher à représenter un texte, il est nécessaire de le travailler de façon à le rendre utilisable par l'algorithme. Voici une série de méthodes et de techniques que l'on applique de manière générale à un texte afin de le rendre utilisable.

5.2.1.1 Tokenization

La *tokenization* consiste dans le découpage du texte en mot appelés *token*. Pour la langue latine, la tokenization est relativement simple puisque les mots sont presque toujours séparés par un espace blanc.

Toutefois quelques difficultés peuvent être rencontrées pour les noms propres à plusieurs mots ou encore à cause de la ponctuation, par exemple le texte "New York-based" doit être découpé en "New York", "based" et non pas en "New" "York-based". Afin de découper les mots de manière correcte, des heuristiques complexes sont utilisés, nécessitant une table où tous les cas spéciaux sont traités.
Voici le résultat d'une tokenization après avoir retiré préalablement toute la ponctuation et laissé des espace blancs à la place :

"Procédés abusifs ,des garanties noyées dans le contrat, non respect des engagements.Désorganisation entre le siège social et les agences.Deux discours différents.Je déconseille il y a mieux."

$\Longrightarrow$

["Procédés","abusifs","des","garanties","noyées","dans","le", "contrat","non","respect","des","engagements","Désorganisation", "entre","le","siège","social","et","les","agences","Deux","discours" ,"différents","Je","déconseille", "il","y","a","mieux"]

Il est aussi utile de transformer toute les lettres majuscules en lettres minuscules.

5.2.1.2 Stop-words

La deuxième étape du preprocessing du texte consiste au retrait des *stop-words* du texte. Les stop-words sont des mots très couramment utilisé de manière générale et qui n'ont pas un sens logique important dans la phrase. En voici certains de langue française : "alors","au","aucun", "aussi","autre","avant", "avec","avoir", "bon", "car", "ce", "cela","ces", "ceux","chaque". Une liste exhaustive se trouve ici [36].

Pour ne conserver que des mots qui ont une plus grande importance (puisqu'ils sont plus rares), on retire l'ensemble des token qui sont des stop-words.

5.2.1.3 Stemming ou racinisation

Cette étape consiste en l'extraction pour chaque token d'une racine. Deux mots, l'un conjugué ou accordé et l'autre dans sa terminologie initiale ne peuvent pas être considérés comme étant similaires pour l'ordinateur puisque leurs orthographes sont différentes. Toutefois, si l'on

36. `http://www.ranks.nl/stopwords/french`

cherche la racine de ces deux mots, celle-ci sera similaire et les deux mots seront alors compris comme étant les mêmes.

La racine d'un mot est obtenue par l'application d'un algorithme de stemming. Il existe différents algorithmes de stemming pour différents langages dont celui de Porter [28] adapté à la langue anglaise à l'origine puis élargi aux autre langages. L'algorithme se base sur des heuristiques pour réduire un mot à sa racine par étape.

Par exemple le mot "GENERALIZATIONS" est transformé en "GENERALIZATION" (Étape 1), puis en "GENERALIZE" (Étape 2), puis en "GENERAL" (Étape 3), et enfin en "GENER" (Étape 4). Le mot "OSCILLATORS" devient "OSCILLATOR" (Étape 1), puis "OSCILLATE" (Étape 2) puis "OSCILL" (Étape 3), puis "OSCIL" (Étape 4).

5.2.1.4 Lemmatization

Cette étape est assez similaire à l'étape précédente. Elle se fait à la place de l'étape précédente. Au lieu de chercher la racine d'un mot, on va plutôt chercher à retirer l'accord de chaque mot. Ainsi les verbes sont mis à l'infinitif et les noms communs sont mis au masculin singulier.

5.2.2 Vectorisation d'un texte

Après avoir appliqué les différentes étapes au texte initial, on obtient une liste de token. Nous allons maintenant chercher à représenter cette liste de token en un vecteur.

5.2.2.1 One-hot encoding

La manière la plus simple est de créer un vecteur de taille *N=Nombre de mots dans le vocabulaire* (c'est à dire nombre de tokens différents) où l'on remplirait la colonne i avec 1 si le i-éme mot du vocabluaire est présent dans le texte (voir l'exemple en figure 56). Cela s'appelle du *one-hot encoding*.

– Text 1: "This is the database lab of the IS master course"
– Text 2: "This is a database course"

	This	is	a	the	database	lab	of	IS	master	course
Text 1:	1	1	0	1	1	1	1	1	1	1
Text 2:	1	1	1	0	1	0	0	0	0	1

FIGURE 56 – Exemple de vectorisation de deux textes, le vocabulaire total est restreint à l'union des ensembles des mots des 2 textes, Michalis Vazirgiannis

Cette vectorisation du document est simpliste puisqu'elle ne prend pas en compte ni la fréquence d'apparition d'un mot ni le contexte dans lequel il apparaît.

5.2.2.2 TF-IDF

Une approche plus développée consiste à représenter un texte (ou document) noté d dans un ensemble de textes (corpus). Pour chaque mot appartenant à un texte, nous allons nous intéresser à sa fréquence d'apparition dans le texte et à sa fréquence d'apparition dans le corpus. Plus un mot apparaît dans un texte, plus le mot est important, mais plus le mot apparaît dans

le corpus moins il est important.

Ainsi on défini la fréquence d'un terme t dans le document d par :

$$tf(t,d) = \text{Le nombre de fois où le terme t apparaît dans le document d}$$

On peut également prendre le logarithme de cette valeur, ou encore le normaliser par la valeur maximum que prend ce coefficient dans un document.

On défini la fréquence d'apparition d'un mot dans un ensemble de documents et on détermine ainsi le poids de chaque terme au sein du corpus. Si on note df_t le nombre de documents où le terme t apparaît, on définit le coefficient idf_t par

$$idf_t = \log_{10} \frac{N}{df_t}$$

où N est le nombre de documents dans le corpus. On passe le coefficient au logarithme afin de réduire son effet sur le résultat final.

Finalement le poids obtenu pour chaque mot t au sein du document d est le suivant

$$w_{t,d} = (1 + log(tf_{t,d})) \log \frac{N}{df_t}.$$

Voici un exemple d'une représentation td-idf d'un corpus :

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth ...
ANTHONY	5.25	3.18	0.0	0.0	0.0	0.35
BRUTUS	1.21	6.10	0.0	1.0	0.0	0.0
CAESAR	8.59	2.54	0.0	1.51	0.25	0.0
CALPURNIA	0.0	1.54	0.0	0.0	0.0	0.0
CLEOPATRA	2.85	0.0	0.0	0.0	0.0	0.0
MERCY	1.51	0.0	1.90	0.12	5.25	0.88
WORSER	1.37	0.0	0.11	4.15	0.25	1.95
...						

FIGURE 57 – Les coefficients tf-idf d'un corpus, en abscisse les termes du corpus, en ordonnée les documents du corpus, Michalis Vazirgiannis

Pour ensuite comparer deux documents, on peut imaginer une distance entre les deux vecteurs tf-idf associés w, w'. La distance euclidienne n'est pas adaptée au sens ou si la taille des documents n'est pas la même alors la distance entre les deux vecteurs est élevée même si ces documents contiennent des informations similaires. Le choix que l'on fait de la métrique dans l'espace de représentation vectorielle des documents est la cosine similarité, c'est à dire le cosinus de l'angle θ entre les vecteurs w, w' donc

$$\cos(\theta) = \frac{< w, w' >}{\|w\| \times \|w'\|}$$

avec l'opérateur $<,>$ étant le produit scalaire.

5.2.2.3 Graphe de mots

Nous avons fait dans les précédentes représentations vectorielles des documents l'hypothèse de l'indépendance des mots présents dans le document. Pour éviter de supposer cette indépendance hypothèse, on introduit les graphes de mots qui vont prendre en compte les relations entre les mots.

A partir d'un texte, on peut construire un graphe où les noeuds sont les mots présents dans le texte et les ponts sont construits en fonction de la succession des mots. Pour un réel k à choisir préalablement, si le mot t suit le mot t' à moins de k mots d'espace alors il existe un pont entre ces deux mots.

Le poids d'un mot dans un document est donc le nombre de voisins qu'il a dans le graphe. En calculant ce poids pour tout les mots d'un document, on retrouve un vecteur de représentation du document.

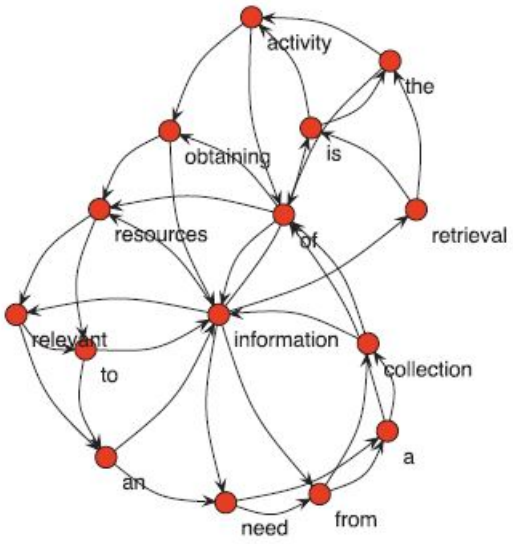


FIGURE 58 – Graphe de mots, Michalis Vazirgiannis

Les graphes de mots peuvent être aussi utilisés dans le cadre de l'extraction de mot clés ou encore pour résumer automatiquement un texte. C'est ce que nous faisons dans la partie 5.3.3.

5.2.2.4 WordtoVec

Les représentations vectorielles des documents vus jusqu'à présent sont basés sur une approche fréquentiste. Une approche davantage basée sur de l'apprentissage automatique existe également. Cette approche se nomme *WordToVec*. Au lieu, de représenter vectoriellement les documents, cette technique permet de représenter vectoriellement les mots. Des techniques d'assemblage existent ensuite pour représenter les documents.

Dans ces articles [24] [25], Mikolov présente des réseau de neurones capable d'apprendre automatiquement une représentation des mots. Mikolov introduit deux structures de réseau de neurones, l'une prédisant un mot sachant son contexte (CBOW) et l'autre prédisant le contexte sachant le mot (Skip-gram). Ces structures sont plus simples que celles d'autres réseaux de

neurones utilisés (voir par exemple [2]) pour la même tâche mais plus efficace et plus rapide à entraîner.

Dans le détail, le modèle *Continuous Bag-of-Words Model* est un réseau de neurones qui a pour but de prédire un mot à la simple connaissance du contexte, c'est à dire des mots entourant le mot à prédire dans un texte. Le modèle est appelé ainsi parce que dans l'utilisation du contexte, le modèle CBOW ne tient pas compte de l'ordre des mots d'où l'image d'un "sac de mots" où les mots ne sont pas rangés. De plus, la représentation obtenu des mots est continu dans l'espace des réels. Voici une représentation graphique du modèle CBOW :

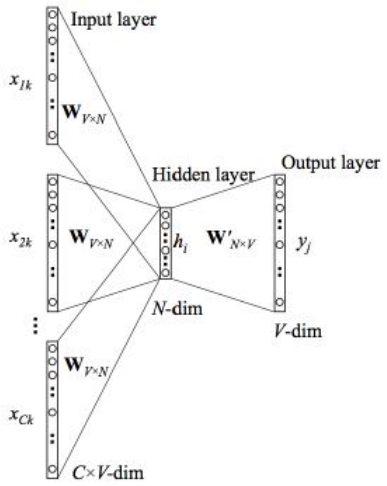


FIGURE 59 – CBOW, Efficient Estimation of Word Representations in Vector Space- Mikolov et al.

C vecteurs de mot obtenus par one-hot encoding sur tout le corpus constitue l'input de ce modèle, C étant la taille choisi du contexte. Les C mots sont les mots appartenant au contexte du mot que l'on veut prédire. On rappelle (voir 5.2.2.1) que la représentation one-hot encoding d'un mot m dans un vocabulaire de taille V donne un vecteur de taille V dont toutes les valeurs sont nulles sauf dans la position correspondant au mot m où la valeur est de 1. Pour entraîner le réseau, l'output que l'on fourni est la représentation one-hot encoding du mot à prédire. Cela signifie en réalité que l'on fait une classification à V classe avec V le nombre de mots dans le corpus.

Le réseau débute par une couche de projection. La projection consiste en une simple matrice de taille $V \times N$ où N est la dimension de la projection que l'on choisit. Chaque vecteur one-hot-encoding représentant les mots du contexte est projeté (multiplié) selon cette matrice et les résultats des projections pour chaque mots du contexte sont sommés pour donner le hidden layer. Finalement, l'output du hidden layer est introduit dans une couche softmax classique à V neurones ce qui donne la probabilité d'obtention pour chaque mot du vocabulaire. La fonction objectif de ce réseau est la fonction objectif classique d'un réseau de neurones terminant par un softmax layer.

Après l'entraînement du modèle, c'est la matrice de projection qui est utilisé comme représentation des mots. Cette matrice a été apprise lors de l'entraînement du modèle par back-propagation. Si $N = 300$ et $V = 10000$, la matrice des poids $W_{V\times N}$ vu sur la figure 59 est la matrice de représentation des mots (voir Fig.60).

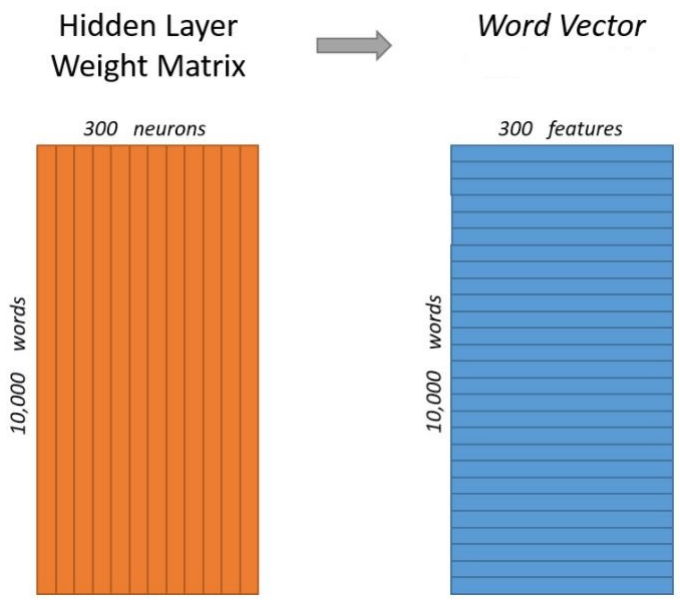


FIGURE 60 – WordToVec [37]

Le modèle Skip-gram se comporte de la même manière que le modèle CBOW, le modèle cette fois-ci tente de prédire le contexte à partir du mot. Voici une représentation du modèle Skip-gram :

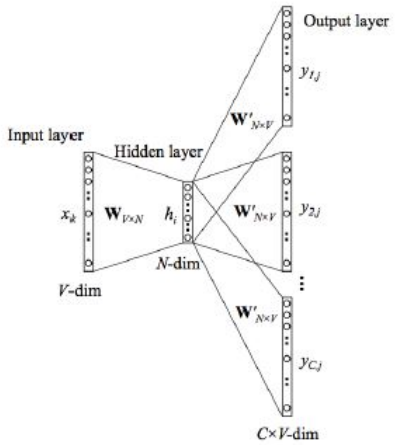


FIGURE 61 – Skip-gram, Efficient Estimation of Word Representations in Vector Space- Mikolov et al.

Ainsi l'input du modèle est la représentation one-hot encoding du mot w_t et l'output est la représentation one-hot encoding du contexte $w_{t+1}, ..., w_{t+C}$. Cette fois-ci la fonction objectif est un peu différente, puisqu'en sortie du réseau, on obtient C vecteurs $(\hat{y}_1, ..., \hat{y}_C)$ de probabilité de taille V (avec C la taille du contexte et V la taille du vocabulaire).

Pour $i, j \in [\![1, C]\!] \times [\![1, V]\!]$,

$$\hat{y}_{i,j} = \mathbb{P}(w_{t+i} = \text{vecteur one-hot-encoding du j-ème mot du vocabulaire}|w_t)$$

37. http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

avec w_t le vecteur one-hot-encoding à l'instant t, le temporalité représente ici la suite de mots que constitue un texte. Ces probabilités sont les probabilités de voir le vecteur one-hot encoding w_{t+i} associé au i-éme mot du contexte être égale au vecteur one-hot encoding du j-éme mot du vocabulaire sachant que l'on connaît w_t. La fonction objectif à optimiser choisie pour entraîner le réseau est alors

$$J = \frac{1}{T} \sum_{1 \leq t \leq T} \sum_{1 \leq i \leq C} \log \hat{y}_{i,l}$$

avec l étant l'indice du vrai mot (connu dans la base d'entrainement) dans le vocabulaire. Cette fonction objectif est bien la log vraisemblance du modèle et c'est celle-ci que l'on va maximiser .

Concrètement ces modèles ont déjà été entraînés par des chercheurs ou des entreprises sur des corpus de tailles gigantesques. L'entraînement des modèles prend plusieurs jours sur des clusters d'ordinateurs. Ainsi, l'entraînement du modèle CBOW ou Skip-Gram sur 6 milliards de mots provenant de Google News prend 2 jours en utilisant 140 CPU (voir [25]). Il est donc préférable de récupérer les représentations vectorielles fournis par Google [38] directement.

5.2.2.5 GloVe

C'est un représentation qui va se baser sur une matrice de co-occurence des mots dans le corpus. Cette matrice de dimension $V \times V$ (V est le nombre de mots dans le corpus) est ensuite factorisée et les dimensions sont réduites de manière à obtenir une matrice de taille $V \times N$ avec N la dimension de la représentation recherchée.

5.2.2.6 ConceptNet

ConceptNet [39] est un graphe logique de mots [34] qui connecte les mots et les phrases avec des ponts labellisés. Le graphe a été conçu avec l'aide de plusieurs sources qui incluent des ressources créées par des experts et des ressources issues du crowd-sourcing (personnes répondant aux questions). Le graphe a été construit de manière à représenter le savoir global nécessaire à la compréhension d'une langue. Cela permet aux algorithmes utilisant le langage humain de comprendre ce que les gens veulent exprimer comme idées.

Ainsi les labels associées à tous les ponts sont du type : *Est utilisé pour*, *Est capable de* , *Est un*. Voici un schéma d'une partie du graphe :

38. `https://code.google.com/archive/p/word2vec/`
39. `http://conceptnet.io/`

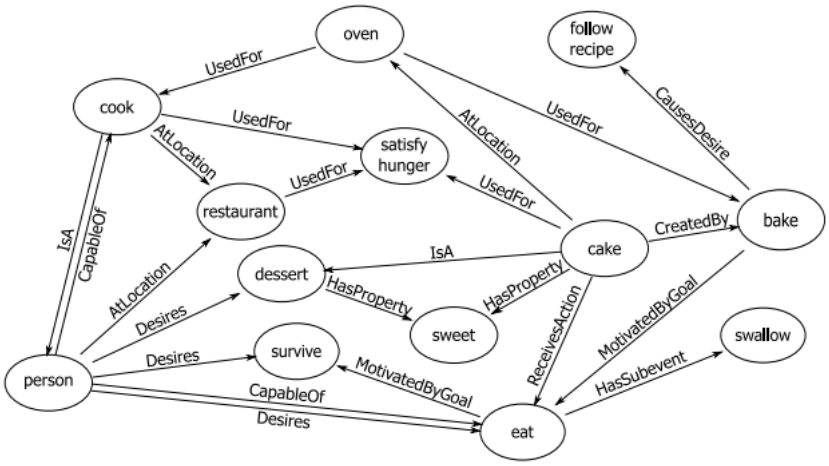


FIGURE 62 – ConceptNet

Lorsque ce graphe est combiné à une ou plusieurs représentations vectorielles des mots, il peut contribuer à améliorer la représentation vectorielle de chaque mot. Ainsi le graphe permet d'assurer un lien logique entre chaque mot. C'est ainsi qu'a été conçu la représentation vectorielle appelée *ConceptNet-Numberbatch*. La description de la méthode se trouve dans l'article suivant [34].

Après avoir concaténé les deux représentations vectorielles WordtoVec et GloVe en une matrice notée w (de taille $V \times 2N$), on applique une régression linéaire contrainte à cette matrice w. La fonction à optimiser de cette régression est la suivante :

$$\phi(q) = \sum_{i=1}^{n} \left[\alpha_i \|q_i - w_i\|^2 + \sum_{(i,j)\in E} \beta_{i,j} \|q_i - q_j\| \right]$$

avec E l'ensemble des ponts appartenant à ConceptNet. Le q optimal subit ensuite une réduction dimensionnelle et donne ainsi une représentation vectorielle des mots.

Concrètement la régression linéaire tend à approcher la représentation vectorielle déjà connu (premier terme de la parenthèse) tout en incitant les mots ayant des liens sur le graphe ConceptNet à avoir une représentation vectorielle proche (second terme de la parenthèse).

Dans la suite, c'est cette représentation que nous utiliserons pour associer un vecteur numérique à un mot.

5.2.3 DocToVec

Pour représenter numériquement un document, il faut trouver un moyen d'associer les différents vecteurs associés aux mots du document. Une manière de faire est de calculer la somme pondérée des vecteurs associés aux mots du document, les poids choisis étant ou uniforme ou bien les coefficients tf-idf des mots dans le document.

Une autre manière d'associer les vecteurs de chaque mot pour obtenir un vecteur associé à une phrase est la suivante. Nous prenons la valeur maximum sur tous les vecteurs associés au mot sur chaque dimension du vecteur. Ainsi, l'agrégation des vecteurs Conceptnet-Numberbacth associés à chaque mot d'un document est transformée en représentation d'un commentaire (voir Fig.63).

	0	1	2	3	4	5	6	7	8	9	...	290	291
Mot													
je	-0.321239	-0.189853	0.115842	0.048545	-0.042594	0.068755	0.019677	0.002223	0.162587	0.062892	...	-0.014659	-0.081595
suis	-0.436238	0.105413	0.110303	0.076845	0.015857	0.047937	-0.001183	-0.043655	0.042164	0.011290	...	-0.009472	0.048106
pas	-0.620610	-0.276561	0.106115	0.081569	0.137261	0.021757	-0.055638	0.004577	0.019408	0.070490	...	0.008388	0.034704
satisfait	-0.274593	-0.115102	0.142340	0.082367	-0.006003	0.117327	0.137636	0.092217	0.125261	0.211025	...	0.020377	-0.003619

	0	1	2	3	4	5	6	7	8	9	...	290	291
Comm_1	-0.274593	0.105413	0.14234	0.082367	0.137261	0.117327	0.137636	0.092217	0.162587	0.211025	...	0.020377	0.048106

FIGURE 63 – Représentation d'un commentaire

On a ainsi la représentation d'un document. Au lieu de prendre le maximum, on peut également prendre le minimum, ou encore le maximum de la valeur absolue. On peut également concaténer les valeurs en prenant les maximuns par colonnes et les minimums par colonnes.

5.2.3.1 Cohérence de la représentation

Pour s'assurer de la cohérence de la représentation des mots, on peut s'intéresser à la distance entre de nombreux couples de mots. On peut également par exemple, réduire la dimension de notre représentation vectorielle par une ACP et représenter différents mots sur les deux composantes principales de l'ACP. Voici le résultat en projetant sur ces axes les couples de mots pays-capitale :

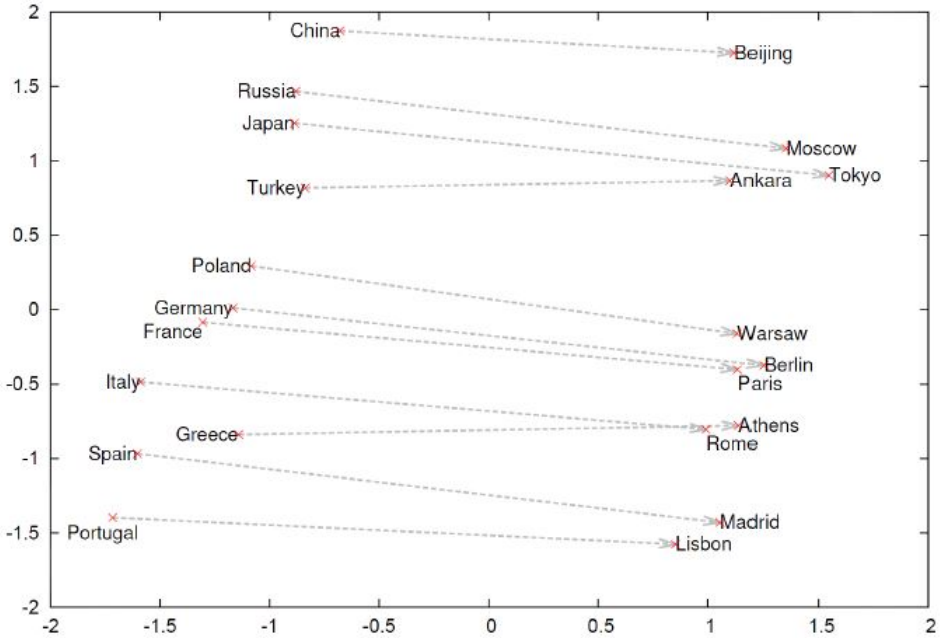


FIGURE 64 – Projection sur les deux composantes principales de l'ACP des représentations vectorielles CBOW des villes et pays, Mikolov et al.

L'alignement des couples nous indique que la représentation vectorielle donne du sens au mot.

5.3 Analyse de réputation

Comment les consommateurs français choisissent leurs assurances ? C'est à cette question que s'est intéressée la sixième édition du baromètre de Promise Consulting Inc [40]. Le constat est alors sans appel, les assurances et les mutuelles ont vu leur réputation se détériorer de manière générale. Traditionnellement, l'engouement pour le secteur assurantiel est faible, "l'assurance est un produit qui relève plutôt de l'obligation et/ou de la commodité, ce qui en fait un secteur peu aspirationnel", note Philippe Jourdan, consologue et fondateur associé de Promise Consulting. A cela s'ajoute la perception des français que la crise économique survenue en 2008 est partiellement causée par AIG, compagnie d'assurance américaine. Ainsi selon le baromètre, les mutuelles bénéficient d'une certaine confiance de la part des français du fait de leur statut : elles relèvent du principe de l'autogestion et poursuivent un but non lucratif, dans l'intérêt de leurs membres.

On peut également ajouter comme élément de réponse à la question posée par le baromètre, la réputation numérique des assureurs. A l'heure des réseaux sociaux et des forums spécialisés, les consommateurs français ont largement la possibilité de commenter leurs expériences auprès de leurs assureurs et d'exprimer clairement leur point de vue sur leurs assureurs. Dans ce contexte, les nouveaux consommateurs de produits d'assurance vont aller chercher des retours d'expérience sur le net et vont avoir tendance à s'orienter vers les assureurs les plus réputés en terme de qualité, de service, de couverture ou encore de prix.

Dans cette partie, nous chercherons à identifier des causes de satisfaction et d'insatisfaction des assurés et analyserons un risque de réputation pour les assureurs. Nous travaillerons ainsi sur des commentaires extraits du site *opinion-assurances* [41].

Cette étude s'est limitée aux compagnies d'assurance et mutuelles, elle pourrait s'étendre à l'analyse d'un risque de réputation pour des entreprises travaillant dans un tout autre domaine [42] [43].

5.3.1 Extraction de données

Les données comprenant des contrats ou des constats sont hautement confidentielles. Nous allons nous restreindre à l'étude de documents publics. Les données que nous extrayons proviennent du site opinion-assurance. Les internautes ont la possibilité de laisser sur ce site un commentaire à propos de leur assurance.

40. http://www.huffingtonpost.fr/2013/02/27/assureurs-preferes-assurances-macif-maif-axa-groupama_n_2772098.html
41. https://www.opinion-assurances.fr/
42. https://www.insuranceinstitute.ca/fr/cipsociety/information-services/advantage-monthly/0517-reputation-risk
43. https://www.insurancespeaker-wavestone.com/2015/05/assurer-le-risque-de-reputation/

FIGURE 65 – Exemple de commentaire sur l'assureur AXA

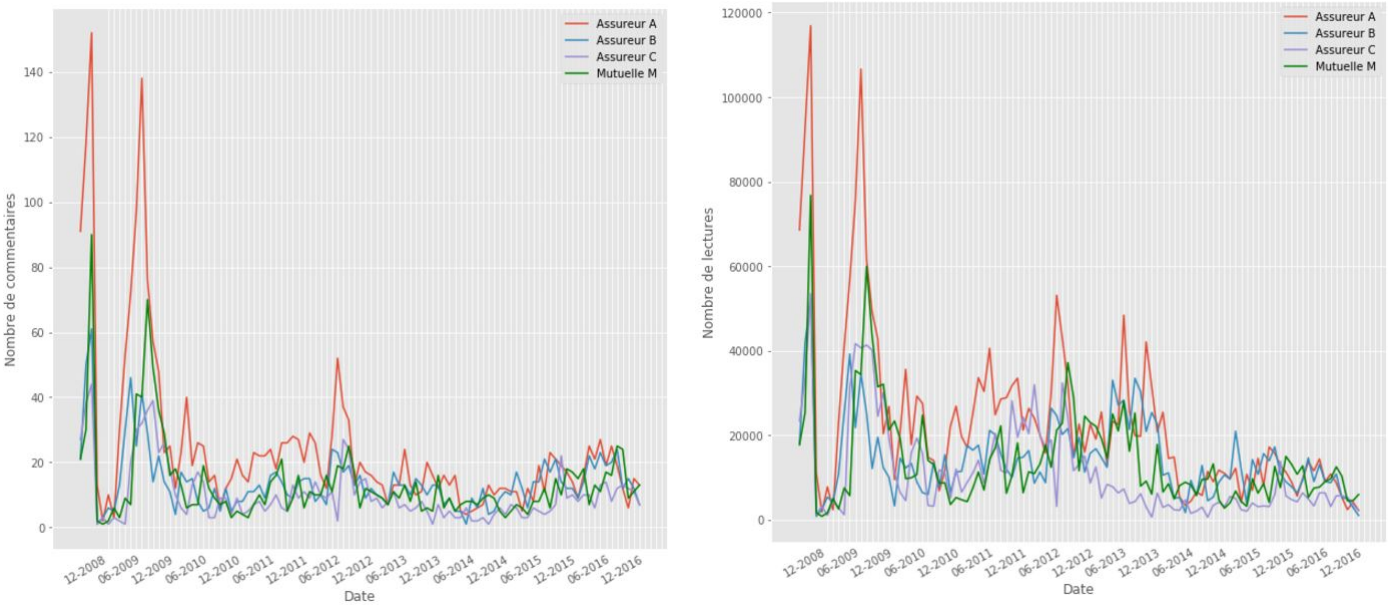
Le commentaire est accompagné d'une notation par étoile sur divers sujet, le prix, la qualité de la garantie, la qualité du service client et la satisfaction générale. Les internautes ont également la possibilité de spécifier les aspects négatifs et les aspects positifs de l'assureur (voir Fig.65). Par ailleurs, le site fourni également le nombre de fois où le commentaire a été lu par d'autres internautes et donne aussi la possibilité aux assureurs de répondre aux internautes.

5.3.2 Statistiques descriptives des données

Pour avoir une étude comparative, nous allons travailler sur 3 assureurs A,B,C et une mutuelle M. La période observée dans laquelle les commentaires ont été postés s'étend de décembre 2008 à juillet 2017. Le nombre de commentaires postés sur le site par mois reste constant après décembre 2009 de même que le nombre de fois où un commentaire est lu (voir Fig.66).

Avant décembre 2009, on peut noter deux pics du nombre de commentaires par mois pour l'ensemble des assureurs et mutuelles. On remarque aussi qu'entre février 2009 et juin 2009, il n'y a eu aucun commentaires postés sur le site. Avec une lecture rapide des commentaires entre décembre 2008 et juin 2009, il nous semble que cela est du à la méconnaissance du site alors. En effet, lors de son lancement en décembre 2008, le site *opinion-assurance.fr* étant encore peu connu, les commentaires sont peu nombreux. Afin de fournir le site en commentaires, ces administrateurs ont du démarcher les assurés lors de campagnes publicitaires ou en faisant des partenariats avec les courtiers en assurances pour qu'ils incitent leurs clients à évaluer leurs assureurs. Les premiers commentaires semblent avoir été faits par des assurés qui ont été démarchés et cela en janvier et en juin. Cela a pu fournir du contenu au site pour inciter d'autres assurés à poster leurs commentaires.

Par rapport aux nombre de lectures, celui-ci reste constant pour chaque commentaire à peu prés. Cela se remarque si les courbes de la figure 66.b correspondent à une translation aux courbe de la figure66.a.



(a) Nombre de commentaires par mois

(b) Nombre de lectures des commentaires du mois

FIGURE 66 – Statistiques descriptives

Entre juin 2012 et décembre 2013 l'intérêt du site est accentué. Cela se voit du décalage entre les courbes 66.a et les courbes 66.b. Cela peut être du aux crises qui ont entraîné une défiance du public envers les assureurs.

Étudions certaines statistiques sur les données de l'ensemble des assureurs. Voici la matrice de corrélations des variables numériques :

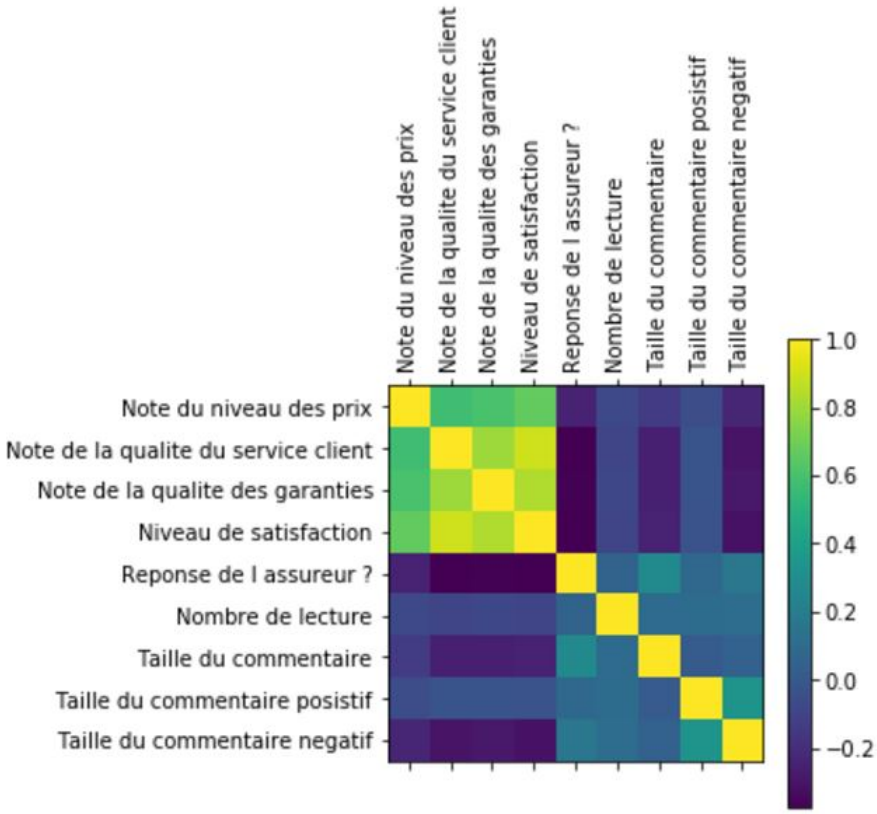


FIGURE 67 – Corrélation entre les variables

La corrélation donne une bonne idée des relations entre les variables. On peut affiner la relation de la satisfaction avec les autres variables par le graphe suivant :

	Niveau_des_prix	Qualit_du_service_client	Qualit_des_garanties	is_response	Nb_of_lecture
Satisfaction					
1	1.715789	1.249474	1.623158	0.367368	1002.895789
2	2.088525	2.340984	2.714754	0.131148	985.013115
3	2.539446	3.285714	3.236674	0.076759	933.825160
4	3.519916	4.287212	4.058700	0.025157	904.790356
5	4.215909	4.882576	4.636364	0.068182	902.606061

FIGURE 68 – Moyenne des variables selon les degrés de satisfaction pour l'assureur A

On remarque ainsi que plus un commentaire traduit une satisfaction faible, plus il est lu mais qu'il fait également plus souvent l'objet d'une réponse de l'assureur. On s'aperçoit également que c'est la "Qualité du service client" qui a le plus gros impact sur l'indice de Satisfaction des gens vis-à-vis de leur assureur et non pas le "Niveau des prix". Ce sont des impressions que l'on retrouve pour l'ensemble des assureurs.

Voici la distribution du nombre de commentaires par degré de satisfaction chez chaque assureur

	Satisfaction 1	% Sat 1	Satisfaction 2	Satisfaction 3	Satisfaction 4	Satisfaction 5	Total
Assureur A	950	38%	305	469	477	264	2465
Assureur B	749	52%	165	229	172	99	1414
Assureur C	466	45%	131	184	163	92	1036
Mutuelle M	536	39%	158	209	235	245	1383

TABLE 4 – Nombre de commentaires par catégories de satisfaction

La mutuelle semble peu épargnée par les critiques par rapport aux compagnies d'assurance. De manière traditionnelle, les mutuelles ont toujours bénéficié d'un à priori positif de la part de leur usagers du fait du but non lucratif de l'organisme. Cela n'est pas évident à la constatation de nos chiffres.

On s'aperçoit également que la taille des commentaires est plus importante lorsque les assurés sont mécontents, l'inverse n'étant pas vrai. Les gens ont plus tendance à expliciter les raisons de leur désagrément que les raisons de leur satisfaction.

5.3.3 Extraction de mots clés et *k-core*

Un des moyens d'analyser les raisons de la satisfaction ou de l'insatisfaction des gens est d'étudier le contenu des commentaires de ces personnes. Afin de résumer de manière concise l'ensemble des commentaires, l'extraction des mots-clés est une méthode couramment utilisée dans l'analyse de textes. L'extraction des mots clés est utilisée partout de nos jours, pour rechercher des informations sur internet, pour trouver des posts similaires, pour faire matcher de la publicité, pour retrouver des articles de recherche, etc.

A partir d'un graphe, on peut déterminer des sous-graphes ayant tous leurs noeuds ayant

un certain degré. Le degré d'un noeud étant le nombre de liens associées à ce noeud. Cet algorithme est appelé *k-core*. On peut ainsi extraire le *core* (sous-graphe) le plus important. Voici une illustration graphique du résultat d'un algorithme :

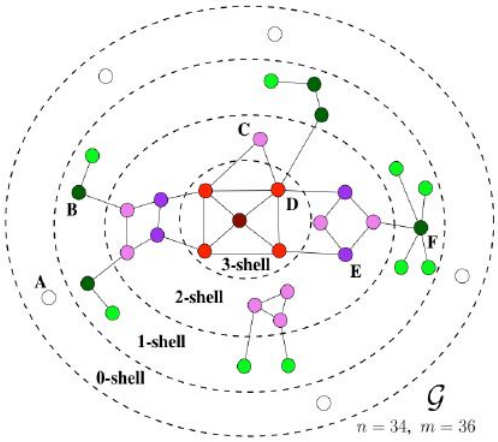


FIGURE 69 – Explication de l'algorithme k-core apliqué au graphe $\mathcal{G}$, le core le plus important est le sous graphe comprenant les noeuds rouges

On construit le graphe des mots associés aux commentaires des personnes ayant une satisfaction de 1 pour l'assureur A, puis on en extrait le sous graphe contenant les mots les plus importants. On rappelle que les mots ont été stemmatisés, et ne sont plus des mots complets.

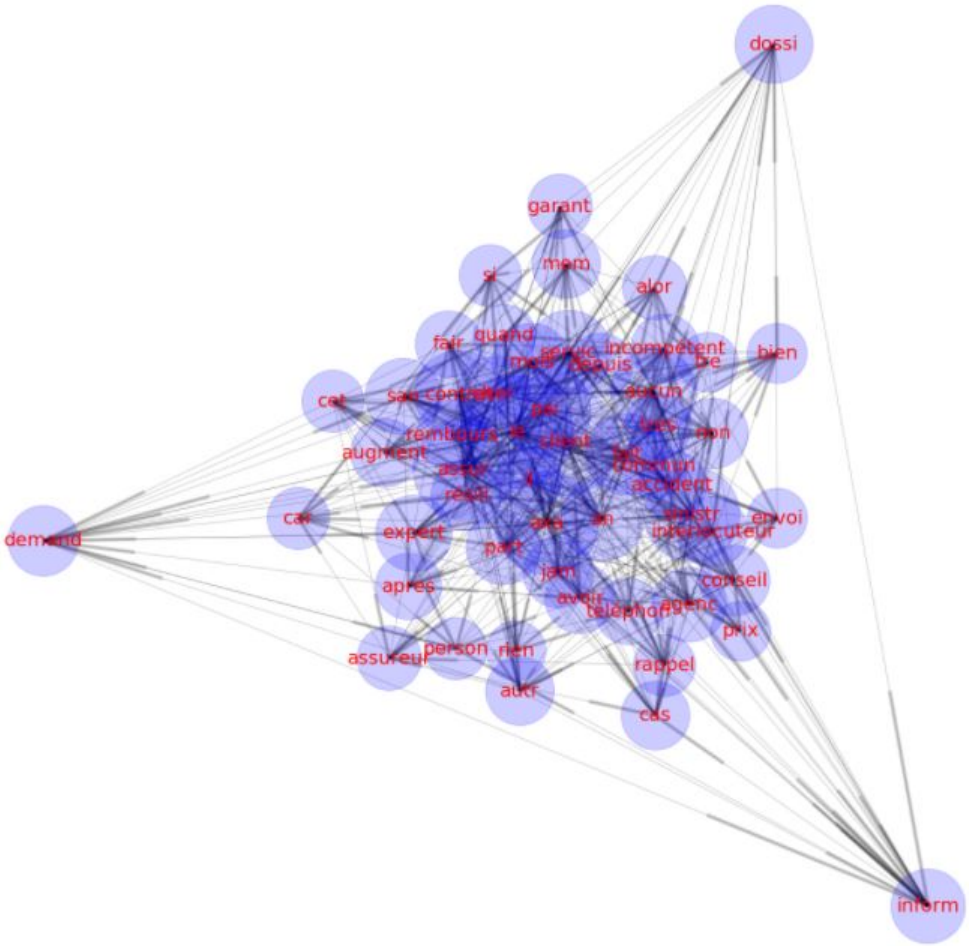


FIGURE 70 – Main core des gens mécontents pour l'assureur A

La géométrie du graphe ne signifie rien. Elle a été choisi de manière à séparer au mieux les noeuds. Malgré cela, visuellement le sous graphe est encore trop grand. Nous allons dons nous intéresser à des parties de celuis-ci. On peut par exemple s'intéresser à certains mots qui expriment une insuffisance. Ainsi on peut récupérer le sous graphe centré autour du moment "aucun"

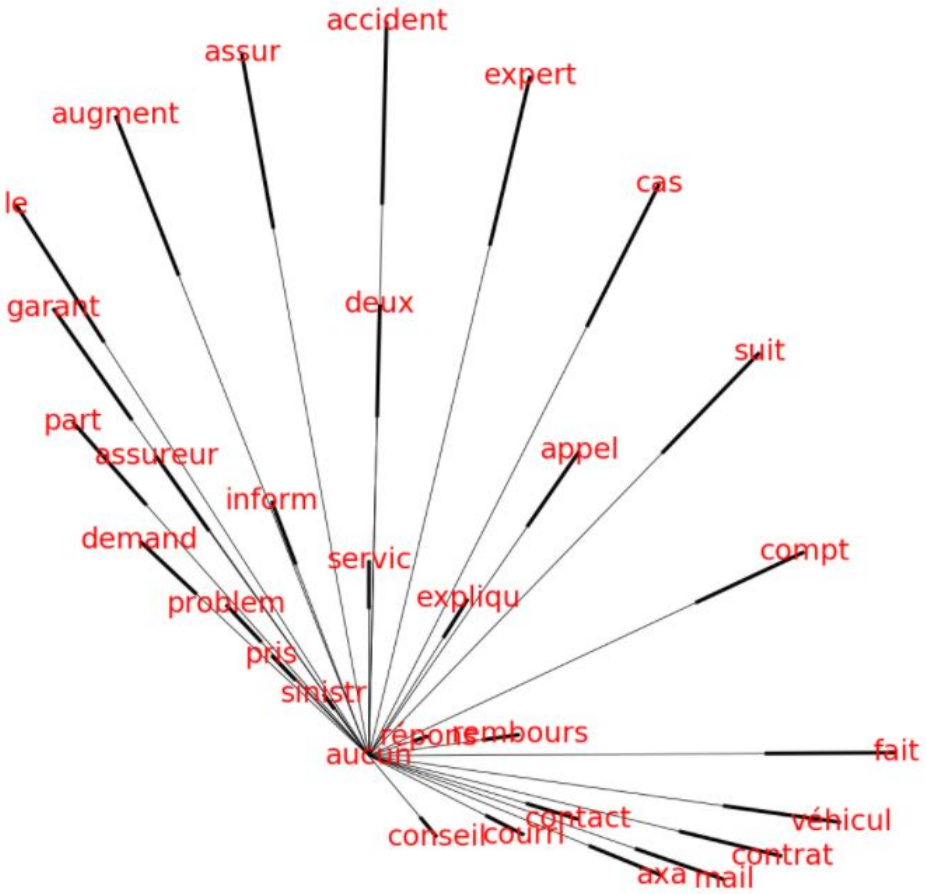


FIGURE 71 – Sous-graphe du main core à partir du mot 'aucun' de l'assureur A

On note ainsi que les gens ont constaté une augmentation de leur prime d'assurance alors que 'aucun'-'accident' (lien présent dans le sous graphe) est un lien fort et revenant souvent dans les commentaires. On voit aussi qu' 'aucun'-'contact', 'aucun'-'conseil', 'aucun'-'mail', 'aucun'-'appel', 'aucun'-'inform' mette en avant les difficultés des individus à joindre leur assureurs. Cela montre et explique la forte corrélation entre la variable de notation de la qualité du service et la variable de satisfaction. On note ici encore une fois donc que plus que le prix, ce qui implique une insatisfaction des gens est le manque de communication et plus généralement une faible qualité de services. On ne parle ici pas forcément de l'assureur mais également des courtiers en assurance qui peuvent ne pas suivre les indications de l'assureur dans le choix de la politique d'après-vente.

5.3.4 Extraction de mots clés par Naïve Bayes

Les méthodes de Naïve Bayes sont des algorithmes d'apprentissage supervisé basé sur le théorème de Bayes avec l'hypothèse naïve que les variables explicatives sont indépendantes entre

elles. Soit une variable catégorielle y et les variables explicatives $x_1, ..., x_n$, le théorème de Bayes spécifie le calcul suivant :

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots x_n \mid y)}{P(x_1, \ldots, x_n)}$$

En utilisant, l'hypothèse naïve suivante $P(x_i|y, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(x_i|y)$, pour tout i cette relation est simplifiée à

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \ldots, x_n)}$$

Puisque $P(x_1, \ldots, x_n)$ est constant étant donné les variables explicatives, on peut utiliser la règle de classification simple suivante

$$P(y \mid x_1, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

On peut alors définir $\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y)$ comme le label prédit en voyant $x_1, .., x_n$, et on peut utiliser l'estimation du Maximum A Posteriori (MAP) pour estimer $P(y)$ et $P(x_i \mid y)$; le premier étant simplement la fréquence relative de la présence du label i dans le data set.

Les différents classifieurs Naïve Bayes diffèrent principalement par l'hypothèse qui est faite sur la distribution $P(x_i \mid y)$. Si on suppose cette distribution gaussienne, cela donne :

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Les paramètres σ_y et μ_y sont estimés par maximum de vraisemblance.

Malgré l'hypothèse trop simple de l'indépendance des variables, les classifieurs Naïve Bayes fonctionnent très bien dans différentes situations, le tri de documents, le filtrage de spam. Il requiert de plus un faible nombre d'exemples en bases de données d'entraînement pour être correctement entraîné.

Au lieu d'utiliser ce classifieur afin de prédire un label, on va l'utiliser à des fins d'extraction de mots clés. En effet, on va utiliser la représentation tf-idf de chaque commentaire comme variable explicative, on a donc $x_1, ...x_{|V|}$ variables avec V l'ensemble des mots présents dans la base de données. On va entraîner le modèle de manière à prédire la classe de satisfaction c'est à dire y =Niveau de Satisfaction de la personne. On va alors analyser les probabilités $P(x_i \mid y = j)$ en fonction des x_i, c'est à dire des mots puis voir quels sont les plus grandes probabilités en fonction de chaque classe.

Puisqu'on a fait l'hypothèse de la gaussianité de ces probabilités, celles-ci sont caractérisées par une moyenne et un variance. On peut donc classer les mots par ordre de grandeur de la moyenne pour chaque classe de satisfaction. Cela nous permet d'avoir concrètement un classement des mots pour chaque classe de satisfaction. Le meilleur mot de la classe 1 est le mot qui va être retrouvé avec la plus grande probabilité si le commentaire est de classe 1.

Niveau de Satisfaction	1	2	3	4	5
Orde d importance					
0	pas	pas	pas	très	très
1	alliant	ans	prix	prix	prix
2	contrat	très	mais	bonne	agence
3	mois	trop	très	pas	cas
4	chez	depuis	cher	cas	agent
5	depuis	mais	ans	bon	bon
6	faire	alliant	bonne	service	écoute

FIGURE 72 – Mots clés extraits par Naïve Bayes par ordre d'importance, 0 étant la plus importante, pour l'assureur B

On constate que les mots "pas" et "très" sont les plus importants selon les classes. Malgré le fait que nous retirons les stop-words, ces mots là sont conservés puisqu'ils ont un sens important, les expressions "satisfaits" et "pas satisfaits" ayant un sens tout à fait opposés. Toutefois un mot seul ne possède pas une signification claire. Il faut l'analyser dans un contexte. C'est pourquoi au lieu d'étudier les mots seuls, nous allons étudier les *bigrams* et les *trigrams*. De manière générale, un *n-gram* est un sous séquence de taille n d'une séquence donnée. Si la séquence est une séquence de mots alors une analyse n-gram est une analyse qui va étudier les suites de n mots.
Ainsi, dans l'analyse tf-idf que nous allons faire, les mots sont remplacés par les suites de deux mots et les suites de trois mots. Le modèle Naïve Bayes reste inchangé et les résultats sont les suivants :

Niveau de Satisfaction	1	2	3	4	5
Orde d importance					
0	service client	déconseille vivement	bonne mais	rapport qualité	satisfaite agence
1	fuir absolument	très déçue	ancien intermédiaire	qualité prix	très bon
2	depuis ans	manque professionnalisme	commentaire moins	rapport qualité prix	qualité prix
3	éviter absolument	plutôt mécontente	globalement correcte	service client	très satisfait
4	mois après	voyez agent	pas années	très bonne	écoute besoins
5	non responsable	après ans	service client	bon service	rapport qualité prix
6	résilié contrat	bonus leurs	chaque année	très bon	aucun problème
7	chez depuis	client depuis	prix compétitifs	rapide cas	très bonne
8	recommande pas	depuis ans	depuis ans	conseille entourage	rapport qualité
9	après avoir	auto habitation	ans sans	adresse recommander	prix très
10	depuis mois	trop chère	année année	bon rapport qualité	disponible écoute
11	toujours pas	après ans résilie	accident juger	bon rapport	prise charge
12	cela fait	ans résilie	jamais accident juger	depuis ans	agent général

FIGURE 73 – Bigrams et trigrams extraits par Naive Bayes par ordre d'importance, 0 étant la plus importante, pour l'assureur B

Cette fois-ci les résultats sont probants. On peut lire la figure 73 de la manière suivante : le bigram "service client" est présent dans le commentaire avec la plus forte probabilité si la satisfaction de la personne ayant écrit ce commentaire est de 1.

En analysant, les différentes classes, mis-à-part les expressions utilisés pour qualifiés l'assureur telle "très bon" pour la classe 5, les raisons de la satisfaction sont assez clairement exposés. Dans la classe 1, nous retrouvons la mauvaise qualité du service client et le temps d'attente trop long qui justifie la déception des personnes appartenant à cette classe. Dans la classe 2, le "manque (de) professionnalisme" est mis avant ainsi que le prix "trop chère". Ces clients note aussi leur fidélité à leur assureurs qui semble ne pas être récompensé ou prise en compte. Dans la classe 3, les gens ont un avis mitigé, leur assurance est "bonne mais"... Dans la catégorie 4, les gens soulignent le "rapport qualité prix" qui semble être très satisfaisant. Finalement dans la catégorie 5, les personnes indiquent avoir un assureur à l'"écoute (de leur) besoins" et des prix très compétitifs.

On peut chercher à réaliser cette analyse, en classant les mots (ou bigrams et trigrams) par ordre de variance. En effet, si la variance de la distribution $P(x_i \mid y = j)$ est grande, cela implique que la distribution est étalée. Les valeurs des coefficients tf-idf associés au mot i sont ainsi distribués dans un large spectre. Cela implique que le mot i a une présence dans les commentaires de classe j qui est largement varié. Dans certains commentaires de type j, le mot i est souvent présent, dans d'autres il l'est moins. Les mots ayants une variance importante peuvent avoir une moyenne faible, ce qui implique que la première analyse ne suffit pas pour les mettre en avant.

Niveau de Satisfaction	**1**	**2**	**3**	**4**	**5**
Orde d importance					
0	fuir absolument	manque professionnalisme	pas années	conseille entourage	satisfaite agence alliant
1	éviter absolument	plutôt mécontente	globalement correcte	adresse recommander	satisfaite agence
2	recommande pas	voyez agent	commentaire moins	malgré prix bonne	utilisé cadre
3	bannir honte	déconseille vivement	ancien intermédiaire	prix bonne	utilisé cadre propriété
4	pas amis	très déçue	bonne mais	pas plusieurs chez	cadre propriété
5	trop proposent	bonus leurs	prix compétitifs	pas plusieurs	agence alliant
6	incompétent éviter	ans résilie	jamais accident juger	plusieurs chez	cabinet sérieux
7	sauve peut	après ans résilie	accident juger	malgré prix	sérieux très conseil
8	supprimée internat	communication sérieuse	bonne mais cher	rapport qualité	sérieux très
9	intervention supprimée internat	pas communication sérieuse	très prix compétitifs	rapport qualité prix	cabinet sérieux très
10	intervention supprimée	après ans	mais cher	rapide cas	très conseil
11	résilié contrat santé	revoir baisse	recommanderai pas compagnie	qualité prix	très satisfait
12	résilié contrat	aucun avantages	leurs mères	bon service	suivi client

FIGURE 74 – Bigrams et trigrams extraits par Naïve Bayes par ordre d'importance en utilisant la variance, 0 étant la plus importante, pour l'assureur B

La lecture simple de la figure 74 est encore assez explicite. Les mots utilisés expliquent parfaitement le degré de satisfaction des personnes. On note également que les individus faiblement

satisfaits vont avoir tendance à décommander leurs assureurs à leurs lecteurs.

5.3.5 Modèle prédictif à but de prédiction de la satisfaction

Le modèle que nous avons vu précédemment est un modèle prédictif utilisé dans le but d'extraire les mots les plus impactant sur les classes de satisfaction. Nous allons essayer maintenant de construire un modèle qui prédit les classes de satisfaction à partir du commentaire. Ce type de modèle peut être efficace dans le but par exemple de catégoriser les mails qui sont adressés aux assureurs : ceux qui contiennent un reproche, ou le motif d'insatisfaction d'un assuré et ceux au contraire des remerciements.

Le problème étudié est donc purement un modèle de machine learning : Prédire la classe de satisfaction d'un commentaire d'assurés. Le score utilisé sera la mesure de précision défini par

$$\text{score} = \frac{\#\text{bonne prédiction}}{\#\text{exemple}}$$

Nous définirons également le problème suivant : est-ce que la classe de satisfaction du commentaire est inférieure ou égale à 2 ? Ce problème est l'équivalent du problème précédent en se ramenant de 5 classes de satisfaction à 2 classes de satisfaction.

Pour résoudre ces problèmes avec un algorithme de machine learning, nous allons construire une base d'entraînement à partir des commentaires d'assurés. Cette base d'entraînement contient des variables explicatives que nous allons expliciter plus loin et une variable à expliquer. Dans le premier problème, la variable à expliquer est la variable de "Satisfaction" des assurés, dans le second problème, la variable à expliquer est la variable de "Satisfaction>=2".

Les variables explicatives de la base d'entraînement utilisée dans l'entraînement des modèles sont constituées en trois étapes. Premièrement, nous prenons la représentation de chaque commentaire selon la technique présenté dans 5.2.3. On choisit de prendre la concaténation des maximums par colonnes et des minimums par colonnes des représentation ConceptNet-Numberbatch de chaque mots. On obtient pour chaque commentaire, un vecteur de taille $2 \times 300 = 600$. Ce sont donc 600 variables explicatives.

Avant d'aller plus loin, nous essayons de détecter directement une structure géométrique des variables explicatives qui diffère selon les degrés de satisfaction des assurés. Cela nous permettra de savoir si une algorithme de machine learning à une chance de séparer les différentes classes. On réalise pour cela une *analyse en composantes principales* (ACP) (voir [10]) des vecteurs que l'on vient de créer. Une analyse en composante principale sur une matrice de type *variables* $\times$ *individus* permet de réduire le nombre de variables en concentrant le maximum d'informations sur les nouvelles variables appelés *composantes principales*. On positionne chaque commentaire selon les deux axes principaux de l'analyse dans le graphe suivant.

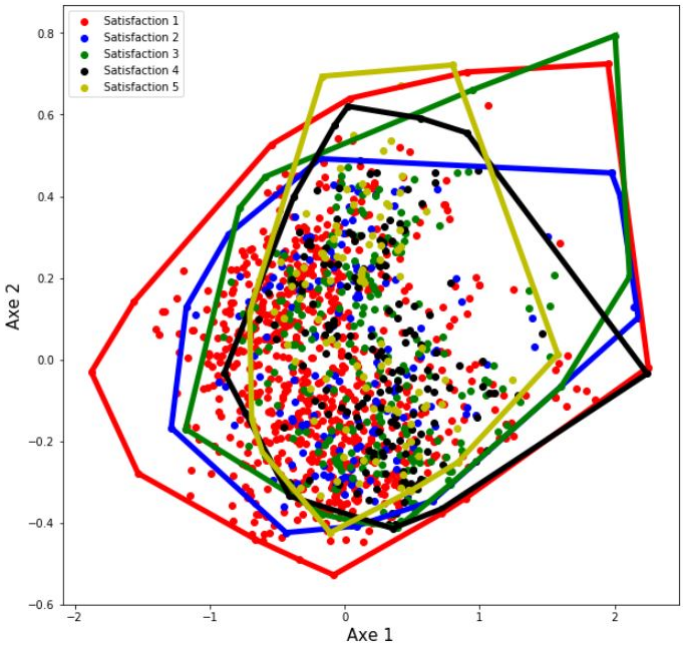


FIGURE 75 – Représentation des commentaires selon les deux axes principaux de l'ACP ainsi que de l'enveloppe convexe des points selon le degré de satisfaction

On voit sur la figure 75 que l'axe 1 explique au moins partiellement le degré de satisfaction puisque selon cet axe, plus la valeur de l'abscisse est grande, moins la satisfaction semble faible. La représentation vectorielle des commentaires des assurés a donc un sens logique. Cela n'est pas une preuve absolue du possible fonctionnement des algorithmes, pour autant cela en donne une intuition.

Jusqu'ici, on a utilisé uniquement le seul texte du commentaire. On peut ajouter comme features 600 variables provenant de la même manipulation appliquée au commentaire positif ("plus", voir comment est constitué le commentaire, 5.3.1) et encore 600 variables pour le commentaire négatif ("moins"). On a finalement un vecteur de taille 1800 pour chaque commentaire.

Nous allons constituer la base train avec les commentaires des assureurs A et C et de la mutuelle M et la base test avec les commentaires de l'assureur B.

Les modèles que l'on va utiliser sont le *gradient boosting* et la régression logistique multi-classe. Ces deux modèles ont démontré leurs qualités sur de nombreuses bases de données et dans des cas très divers.

Accuracy...	Gradient Boosting	Régression logistique
... du problème à 5 classes	**0.618**	0.587
... du problème à 2 classes	**0.834**	0.815

TABLE 5 – Résultats de la prédiction

L'accuracy sur le premier problème est assez faible. En effet, il semble que la classe 2 soit très rarement prédite et à la place, c'est la classe 1 qui est prédite. La matrice de confusion qui croise le vrai label et le label prédit nous le confirme. La matrice se lit de la façon suivante : dans la case en haut à gauche, un grand nombre (se voit sur l'échelle) de prédictions sur les commentaires sont à 1 alors que le vrai label du commentaire est de 1. De même pour chaque

case. Une classification parfaite donnerait des valeurs importantes sur la diagonale et nulles ailleurs.

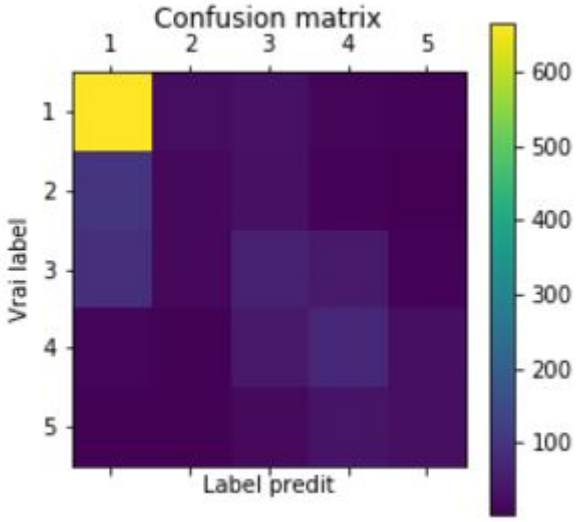


FIGURE 76 – Matrice de confusion en utilisant le Gradient Boosting

Dans notre problème, sur la matrice de confusion (voir Fig.76), on voit que les cases sur la diagonale et proches des diagonales ont des valeurs importantes tandis que les valeurs sont plus faibles ailleurs. Cela indique que la prédiction n'est pas parfaite mais que dans l'ensemble, les erreurs lourdes (prédiction de 1 alors que le vrai label est de 4 ou de 5 ou l'inverse) sont peu nombreuses. C'est également ce qu'indique les résultat du second problème sur la table 5. On a en effet, réussi à prédire à 82% si un commentaire est écrit par une personne de satisfaction 1 ou 2.

De plus, ces résultats ne sont pas biaisés du fait des proportions globales des classes visibles en table 4. On a à peu prés 50% de commentaires de satisfaction de classe 1-2 et 50% de classe 3-4-5 pour les assureurs A,C et M. Si ces proportions avaient été de l'ordre 90% contre 10%, un biais important serait introduit dans les prédictions. On peut donc confirmer que les résultats de la prédiction de la satisfaction d'un assuré à partir de son commentaire sont bons.

Ainsi, nous avons pu calibrer un modèle pouvant à partir du commentaire d'un assuré, déterminer sa satisfaction. Ce type de modèle peut être utilisé dans l'analyse automatique de mail ou encore dans l'élaboration de "Chatbot" pour la création d'aide automatique envers les assurés. De manière plus général, l'analyse de la réputation d'un assureur pourrait utiliser ce modèle en déterminant le degré de satisfaction de tous les posts, commentaires, réactions présents sur Internet, dans les médias, etc. Ensuite, une analyse statistique sur les résultats donnerait une idée de la réputation de l'assureur.

5.3.6 Définition et évaluation du risque de réputation, création d'un indicateur sur la réputation

Les analyses que nous avons faites précédemment (sauf la prédiction) se basent sur l'utilisation des commentaires des différents groupes séparément. Il peut également être utile de comparer les assureurs entre eux pour pouvoir définir un indicateur sur la réputation de l'assureur relativement aux autres assureurs.

D'après la définition du risque opérationnel vu dans la première partie du mémoire (voir 2.1),

celui-ci n'inclut pas de risque de réputation. Ce risque étant beaucoup plus difficile à évaluer et à prendre en compte, il fait l'objet d'un faible nombre d'études et est très mal estimé. Ainsi selon les experts d'*Aon Risk Solutions*, « nombreux sont ceux qui estiment que les entreprises négligent le préjudice à l'image de marque et à la réputation jusqu'à ce qu'une crise éclate » [44]. Selon le *Reputation Institute* de New-York, la réputation d'une banque repose sur sept leviers ([26],[5]) :

- les produits et services : qualité des produits, rapport qualité/prix, adéquation aux attentes,
- la gouvernance : ouverture et transparence de l'entreprise, éthique des affaires,
- la citoyenneté : responsabilité sociale et environnementale, soutien de causes,
- l'emploi : rétribution juste et équitable des salariés, bien-être des collaborateurs,
- l'innovation : capacité d'innovation, capacité d'adaptation aux changements du marché,
- le leadership : crédibilité et charisme des dirigeants, qualité du management, vision du futur,
- la performance : profitabilité, perspectives de croissance.

Ces sept leviers se retrouvent tous dans le définition du risque de réputation des compagnies d'assurances. Notre étude s'intéresse particulièrement au premier levier puisque l'avis des consommateurs se porte de manière générale, uniquement sur cet aspect.

L'article suivant [33], qui s'intéresse à la réputation des établissements bancaires, note que " l'évaluation de la valeur de la réputation en tant qu'actif immatériel et levier de succès est particulièrement délicate. Son entretien et son développement demande une attention de tous les instants et par conséquent une veille informationnelle constante. Le manque de connaissances des sources potentielles de menaces sur la réputation nécessite des études préalables des concurrents et une veille permanente des différents réseaux de flux d'information (presse, Internet. . .)."

Notre étude se limite à des données issues de commentaires de personnes sur un forum spécialisé. D'autres données telle que la presse seraient à utiliser afin de prolonger l'étude du risque de réputation. Afin d'évaluer tout indicateur sur le risque de réputation, il est nécessaire de s'interroger sur les conséquences de ce risque. La réputation d'une entreprise impacte sur le résultat de celle-ci. Une entreprise ayant une bonne réputation a davantage d'attrait pour les clients. De plus, confronté à l'entrée en vigueur de la loi Hamon qui assouplit la procédure de résiliation des contrats d'assurance (désormais possible à tout moment après un an de contrat), les assureurs font face à des clients de plus en plus volatiles, ce qui rend l'impact de la réputation encore plus important.

Afin d'améliorer le taux de rétention de ces clients, un assureur peut suivre ces assurés et leur comportements afin d'anticiper leur départ et activer un processus de rétention. C'est ce qu'a débuté la Maif [45] en utilisant les données issues des échanges avec ces client avec des méthodes de text mining.

Nous proposons une autre approche, plus globale, qui cherche à estimer la réputation d'un assureur de manière général dans l'opinion publique. Encore une fois, les données de souscription étant extrêmement sensible, nous nous limiterons à la proposition d'indicateurs sur la réputation des assureurs.

44. http://www.argusdelassurance.com/risk-management/prejudice-a-l-image-de-marque-et-a-la-reputation-
98979

45. http://www.e-marketing.fr/Thematique/general-1080/Breves/Comment-Maif-anticipe-evite-depart-ses-cl
htm#isVWvBZ9XHQbThU7.97

Concrètement, afin de suivre la réputation d'un assureur simplement, on peut s'intéresser à la satisfaction de ces clients dans le temps. Voici l'évolution de la moyenne de la satisfaction des assurés dans le temps pour les mêmes assureurs que ceux choisis précédemment.

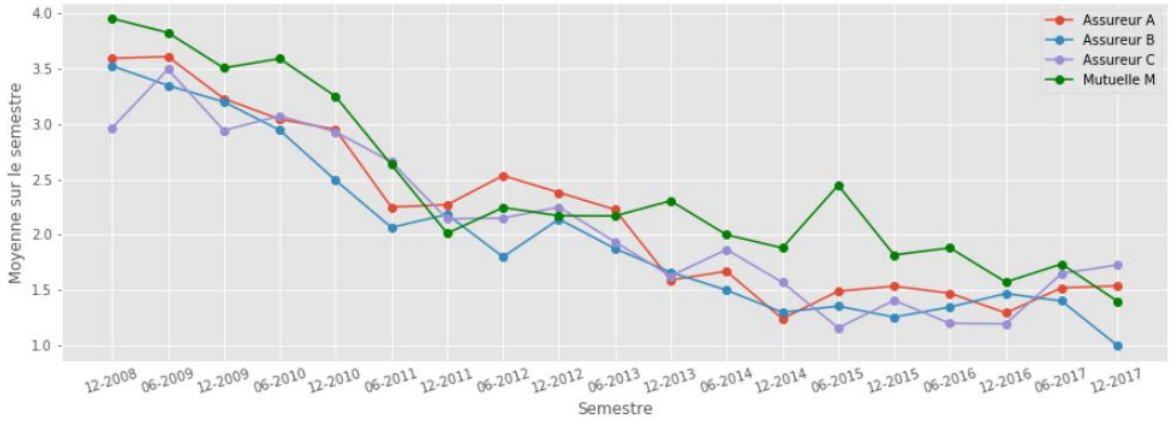


FIGURE 77 – Moyenne de la satisfaction des assurés par semestre

C'est notre premier indicateur sur la réputation des assureurs. Sur la figure 77, on voit la diminution de la satisfaction des gens sur 19 semestres entre 2008 et 2017. Cette diminution concerne l'ensemble des assureurs et avec une forte pente dans les années 2011-2012. La mutuelle semble peu épargnée par la diminution de la moyenne de la satisfaction des gens.

Un second indicateur de la réputation des assureurs peut s'intéresser directement à la "négativité" des commentaires. En s'intéressant aux contenus des commentaires, on s'aperçoit que le niveau d'insatisfaction des personnes est visible. De plus, les commentaires des personnes mécontents sont de manière générale ceux qui sont davantage lues. On peut donc créer un indicateur de la réputation des assureurs en estimant la négativité des commentaires.

Pour ce faire, on constitue un ensemble de commentaires dont la satisfaction associée est de 1 et postés avant 2011 noté A. On va s'intéresser à la similarité du contenu des commentaires entre 2012 et 2017 et A. La similarité entre un commentaire m et A est calculé par la somme des cosine similarités entre m et les commentaires de A.

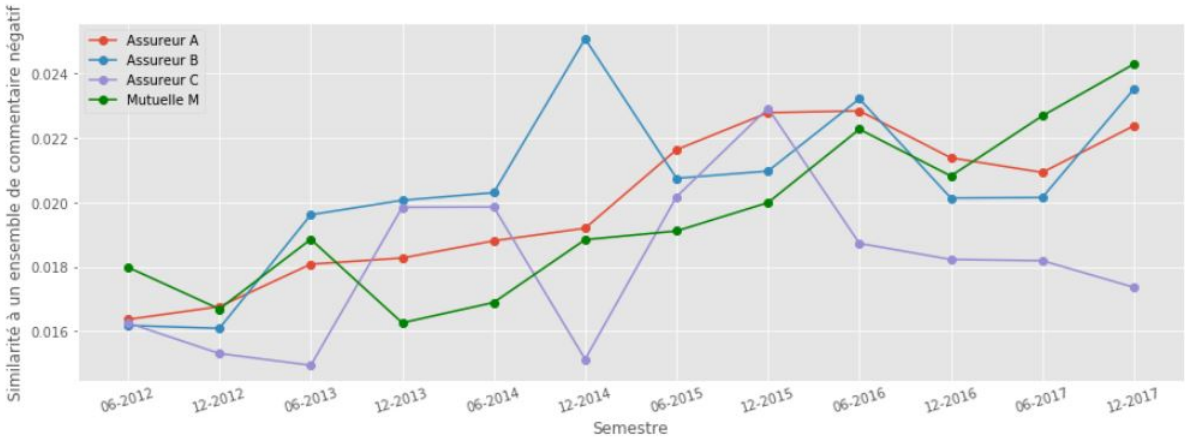


FIGURE 78 – Moyenne de la similarité des messagesà l'ensemble A par semestre

On remarque sur la figure 78 que la mutuelle M a des commentaires qui s'apparente de plus en plus à des commentaires dont la satisfaction est faible. Au contraire, l'assureur C a une tendance à la baisse de l'indicateur et est loin des autres assureurs dans la période la plus récente.

Cela était moins flagrant au niveau de l'indicateur précédent. De plus, on note en 2014 une dégradation de l'indicateur pour l'assureur B. Cela correspond à la dégradation de la moyenne pour l'assureur B en 2014.

Il est très difficile de lier ces tendances directement à l'action des assureurs. Cependant le constat que les assureurs sont de moins en moins biens perçu dans l'opinion publique se retrouve globalement dans nos résultats. A partir de ces indicateurs, il serait intéressant d'étudier le nombre de souscription et de résiliation par années. Toutefois ces données sont confidentiels et il est impossible d'en avoir un aperçu à partir des comptes de résultats des assureurs.

6 Conclusion

Dans ce mémoire, nous avons présenté et implémenté les modèles permettant d'extraire du texte sur des documents scannés. Bien qu'ayant obtenu des résultats perfectibles, les algorithmes basés sur le Deep Learning ont prouvé leur efficacité sur des images de bonne qualité. L'amélioration du réseau de neurones proposé pour la reconnaissance de caractères manuscrits se fera en entraînant le modèle sur des bases de données d'entraînement plus diversifiées. Pour l'écriture tapuscrite, les résultats obtenus grâce à des algorithmes libres d'accès sont de bonne qualité. Il est toutefois nécessaire de les comprendre pour en connaître les limites et notamment les cas d'utilisation où ceux-ci ne pourront obtenir de bons résultats.

L'analyse des commentaires des assurés, au moyen des méthodes du text-mining, a permis de mettre en évidence la possibilité de détecter un sentiment chez les assurés à travers leurs écrits. Il a ainsi été possible de lire automatiquement des milliers de commentaires et d'en extraire les raisons du niveau de satisfaction des internautes vis-à-vis de leurs assureurs. Nous avons ainsi construit deux indicateurs sur le niveau de réputation des assureurs. Ces modèles et indicateurs pourront ensuite être élargis et devenir exhaustifs en analysant les réseaux sociaux, les mails ou tout autre message adressé à l'assureur.

7 Annexe

7.1 Tesseract

Tesseract a été développé, à l'origine, par les laboratoires de Hewlett-Packard à Bristol et à Greeley dans le Colorado entre 1985 et 1994 avec quelques changements faits pour adapter la technologie à Windows en 1996 et la traduction du code initiale en code C++ en 1998. En 2005, Tesseract a été rendu Open-Soucre par HP. Depuis 2006, son développement est assuré par Google (voir Fig.79).
La dernière version stable a été délivrée le 1er Juin 2017 et l'intégration de réseaux de neurones dans cette technologie est déjà prévue, une version alpha de ce prototype a été délivrée.

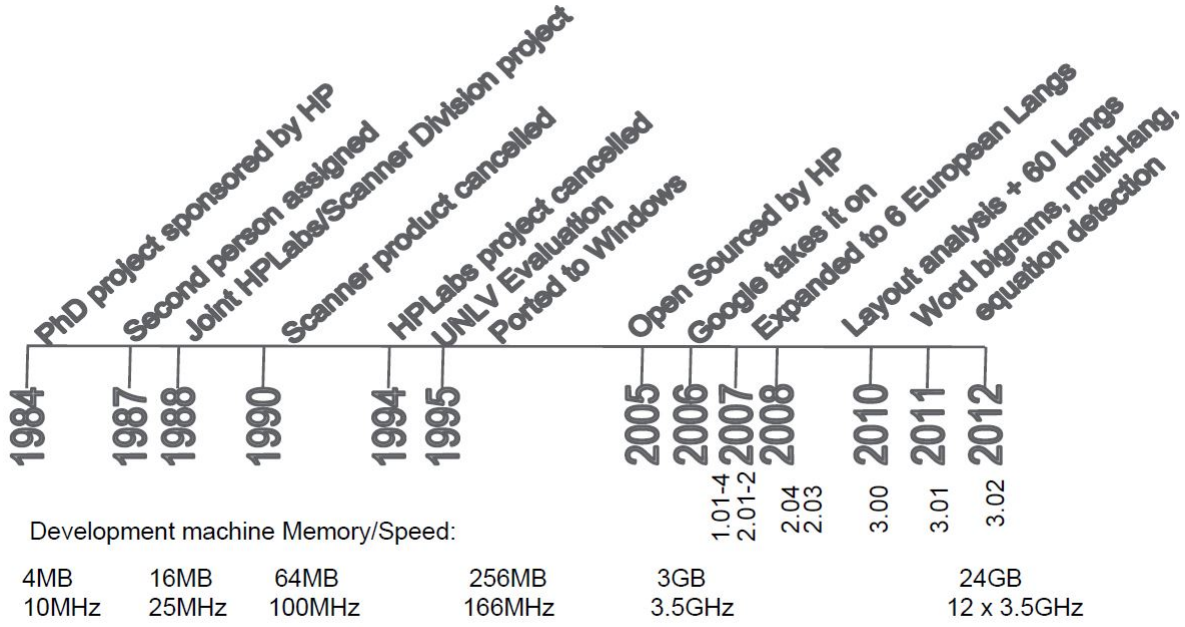


FIGURE 79 – Historique de l'évolution de Tesseract, Ray Smith, DAS 2014

7.2 Modèle HMM

Lorsque l'échantillon sur lequel on travaille ne peut admettre l'hypothèse IID, il faut étudier d'autres modèles pour modéliser la distribution de ces données. Les modèles HMM sont adaptés pour modéliser les données séquentielles, c'est à dire des données successives que l'on ne peut pas considérer comme indépendantes.

Les HMM sont des généralisation de modèles de mélange. Ces modèles peuvent être décrits de la manière suivante : pour générer un point selon un modèle de mélange, il suffit de générer "l'état" dans lequel se trouve ce point puis de générer le point étant donné la distribution de l'état.

Dans les modèles HMM, on ne suppose plus que les états soient choisis indépendamment au cours du temps, on va au contraire supposer que l'état choisi dépend de l'état précèdent. On introduit alors une matrice de transition qui modélise le passage d'un état à l'état suivant selon des probabilités. Cette matrice de transition étant constante dans le temps, la propriété markovienne de la chaîne est assurée. Le reste est inchangé par rapport à un modèle de mélange.

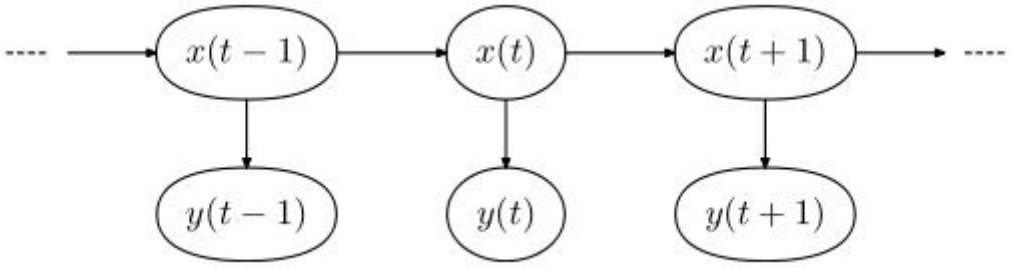


FIGURE 80 – Modèle HMM

7.2.1 Calcul d'inférences

Le problème d'inférence pour le modèle HMM est donc le suivant : étant donné une séquence de données observées, il faut estimer les distributions de probabilités des différents états et les probabilités de transitions dans les différents états.

On peut représenter le modèle HMM comme une chaîne de variables aléatoires. Une des conséquences du modèle graphique ainsi défini est que "le futur est indépendant du passé étant donné le présent". Si on note x_t l'état à l'instant t et y_t est l'output observable, on a alors : conditionnellement à x_t, x_{t-1} et x_{t+1} sont indépendants, autrement dit conditionnellement à la connaissance de l'état présent, l'état futur et l'état passé sont indépendants. Attention, la connaissance du présent ne peut se résumer à la connaissance de y_t, il faut connaître x_t.

La paramétrisation du modèle donne les paramètres suivants : le premier état est choisi selon une distribution π. Si on possède M états différents, π est de dimension M. On a donc $\pi^i = p(x_0^i = 1)$. Puis la matrice de transition A est de taille $M \times M$ avec pour chaque entrée de la matrice $a_{i,j} = p(x_{t+1}^j | x_t^i)$. Les derniers paramètres concernent la distribution $p(y_t|x_t)$. On peut ensuite grâce aux indépendances conditionnelles du réseau, calculer la probabilité jointe

$$p(x, y) = p(x_0) \prod_{t=0}^{T-1} p(x_{t+1}|x_t) \prod_{t=0}^{T} p(y_t|x_t)$$

Le principal problème d'inférence qui découle du modèle HMM est l'inférence de la probabilité d'émission de la séquence d'une chaîne d'état x étant donnée une séquence de valeurs observée y $p(x|y)$. Il peut être aussi utile de ne s'intéresser qu'à l'état x_t ou encore $p(x_t|y_0, ..., y_t)$ (filtering problem) ou encore $p(q_x|y_0, ..., y_s)$ où $t > s$ (prediction problem) ou encore $p(x_t|y_0, ...y_u)$ où $t < u$ (smoothing problem).

Par définition, le calcul de $p(x|y) = p(x, y)/p(y)$ nécessite le calcul de $p(y)$. Faire ce calcul implique de faire une somme sur l'ensemble de valeurs possibles des état cachés x :

$$p(y) = \sum_{x_0} \sum_{x_1} ... \sum_{x_T} \pi(x_0) \prod_{t=0}^{T-1} a_{x_t, x_{t+1}} \prod_{t=0}^{T} p(y_t|x_t) \tag{1}$$

Cette somme ne peut se résoudre directement puisqu'il s'agit d'un calcul à M^T opérations ce qui est considérable pour des valeurs de M, T raisonnables.

On va donc reformuler le problème pour que le calcul soit réalisable de manière récursive. On a

$$\begin{aligned}
p(x_t|y) &= \frac{p(y|x_t)p(x_t)}{p(y)} \\
&= \frac{p(y_0, ..., y_t|x_t)p(y_{t+1}, ..., y_T|x_t)p(x_t)}{p(y)} \\
&= \frac{p(y_0, ..., y_t, x_t)p(y_{t+1}, ..., y_T|x_t)}{p(y)} \\
&= \frac{\alpha(x_t)\beta(x_t)}{p(y)}
\end{aligned}$$

De plus en sommant sur toutes les valeurs que peut prendre x_t, on doit obtenir la valeur de 1, on peut donc assurer $p(y) = \sum_{x_t} \alpha(x_t)\beta(x_t)$. Nous avons donc réduit le problème au calcul des valeurs α et β. Ces quantités peuvent se calculer de manière récursive car

$$\begin{aligned}
\alpha(x_{t+1}) &= p(y_0, .., y_{t+1}, x_{t+1}) \\
&= p(y_0, .., y_{t+1}|x_{t+1})p(x_{t+1}) \\
&= p(y_0, .., y_t|x_{t+1})p(y_{t+1}|x_{t+1})p(x_{t+1}) \\
&= p(y_0, .., y_t, x_{t+1})p(y_{t+1}|x_{t+1}) \\
&= \sum_{x_t} p(y_0, .., y_t, x_t, x_{t+1})p(y_{t+1}|x_{t+1}) \\
&= \sum_{x_t} p(y_0, .., y_t, x_{t+1}|x_t)p(x_t)p(y_{t+1}|x_{t+1}) \\
&= \sum_{x_t} p(y_0, .., y_t|x_t)p(x_{t+1}|x_t)p(x_t)p(y_{t+1}|x_{t+1}) \\
&= \sum_{x_t} p(y_0, .., y_t, x_t)p(x_{t+1}|x_t)p(y_{t+1}|x_{t+1}) \\
&= \sum_{x_t} \alpha(x_t)a_{x_t,x_{t+1}}p(y_{t+1}|x_{t+1})
\end{aligned}$$

Ainsi pour calculer les α, le temps requis sera en $O(M^2T)$. A noter que la récurssion se fait en avant dans le temps. De même, on peut démontrer la relation

$$\beta(x_t) = \sum_{x_{t+1}} \beta(x_{t+1})a_{x_t,x_{t+1}}p(y_{t+1}|x_{t+1})$$

qui donne une récursion "en arrière" dans le temps. La méthode est appelée *alpha-beta récursion* ou encore *forward-backward algorithm*. Il existe d'autres méthodes similaires pour calculer $p(y)$ par exemple *gamma récursion* et donc faire de l'inférence.

7.2.2 Estimation des paramètres

A part l'inférence, il est nécessaire d'estimer les paramètres du modèle HMM à partir des données que l'on possède. On pose θ étant l'ensemble des paramètres du modèle, la vraisemblance est donnée par $p(y|\theta)$ pour une séquence observée y. En prenant la log vraisemblance du modèle, on a

$$p(y|\theta) = \log \sum_{x_0} \sum_{x_1} ... \sum_{x_T} \pi(x_0) \prod_{t=0}^{T-1} a_{x_t,x_{t+1}} \prod_{t=0}^{T} p(y_t|x_t).$$

Notre but sera de maximiser cette valeur par rapport à θ. Selon la distribution que l'on définit pour la séquence observable $p(y_t|q_t)$, on trouve des formules fermées pour chaque étape de l'algorithme EM qui nous permet de maximiser cette log vraisemblance.

7.3 Résultats d'une segmentation

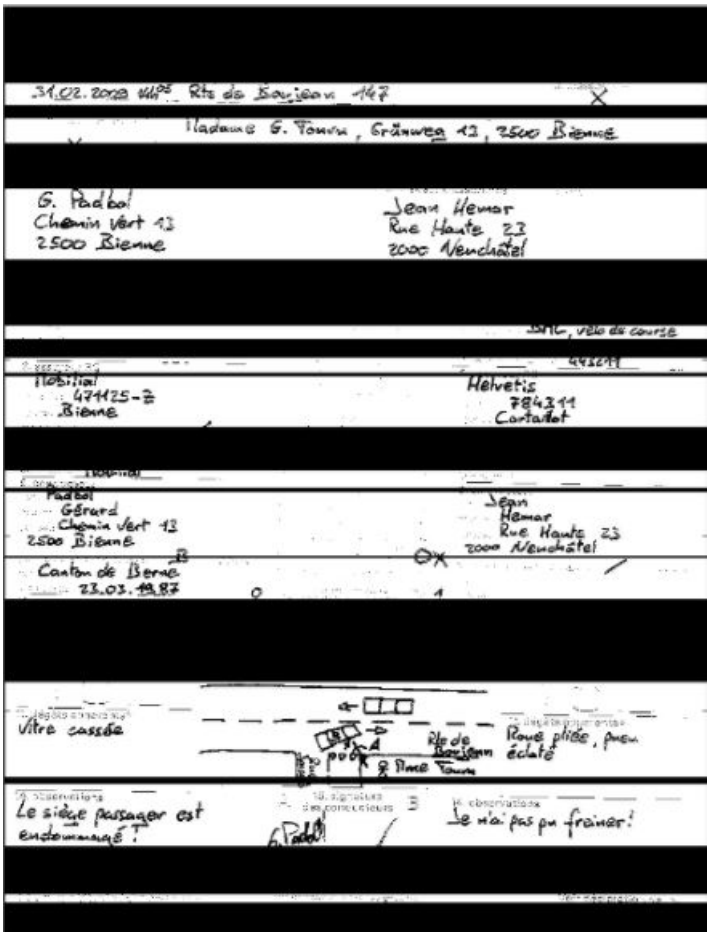


FIGURE 81 – Constat segmenté

7.4 Nombre de paramètre à optimiser

La première couche du réseau est une couche de convolution. Les seuls paramètres à apprendre sont les valeurs de la matrice de convolution de taille 3×3 et 1 paramètre que l'on associe dans la somme (voir 3.6.3.1). Étant donné que l'on fait 16 convolutions en parraléle, cela donne le nombre de paramètres suivant :

$$160 = (3 \times 3 + 1) \times 16$$

La convolution suivante est la parallélisation de 16 convolution 3D (similaire à la 2D, le matrice de noyau ayant trois dimensions), le nombre de paramètres est

$$2320 = (3 \times 3 \times 16 + 1) \times 16$$

La couche dense suivante prend en entré une matrice de taille 32×256 et renvoie une matrice de taille 32×32. Cela implique une matrice de poids de taille $(256 + 1) \times 32$ (voir le feedforward network). Le nombre de paramètre de cette couche est donc

$$8224 = (256 + 1) \times 32$$

Une couche GRU fait intervenir trois fois deux matrices et un vecteur ($W_z, U_z, b_z, W_r, U_r, b_r, W_h, U_h, b_h$ dans la définition de la couche GRU 3.6.3.2). On a décidé que la couche renverrait un output de taille celle de la séquence $\times 512$ donc les matrices W sont de tailles 32×512, les matrices U de taille 512×512 et les vecteurs b de taille 512×1. Le nombre de paramètres pour la couche GRU (forward) est donc

$$837120 = (32 \times 512 + 512 \times 512 + 512) \times 3$$

De même la couche GRU (backward) possède 837120 paramètres. Les couches GRU suivantes (forward et bacward) possèdent chacune

$$1574400 = (512 \times 512 + 512 \times 512 + 512) \times 3$$

Finalement la couche de sortie prend en entrée une matrice de taille 1024×32 et renvoie une matrice de taille 65×32. La matrice des poids possède donc

$$66625 = (1024 + 1) \times 65 \text{ paramètres}$$

Le nombre totale de paramètre du réseau de neurones que nous avons entraîné est donc

$$160 + 2320 + 8224 + 2 \times 837120 + 2 \times 1574400 + 66625 = 4900369$$

7.5 CPU contre GPU

Contrairement aux autres algorithmes de machine learning, les réseaux de neurones utilisent plus souvent des cartes graphiques (*GPU*) pour réaliser l'ensemble des calculs que des processeurs (*CPU*). Traditionnellement, les réseaux de neurones étaient entraîné sur des CPU sur une seule machine. Cela est considéré comme largement insuffisant aujourd'hui. L'entraînement des réseaux de neurones se fait aujourd'hui en utilisant des cartes graphiques combinés à des multiples processeurs, le tout mis en réseau. Avant de créer ces machines, les scientifiques ont travaillé dur afin de démontrer que les CPUs ne peuvent gérer la charge de calcul très importante demandé par les réseaux de neurones.

Les GPUs pour *Graphics processing units* (unité de traitement graphique) sont des composantes matériels spécialisés développés à l'origine pour les applications graphiques. Le marché en plein développement des jeux vidéo a stimulé ce développement de matériel graphique de haute qualité. Les caractéristiques de performance requises pour de bons systèmes de jeux vidéo sont précisément celles pouvant être bénéfique pour les réseaux de neurones aussi.

Le rendu du jeu vidéo nécessite d'effectuer de nombreuses opérations en parallèle rapidement. Les modèles de personnages et d'environnements sont spécifiés en termes de listes de coordonnées 3-D et de sommets. Les cartes graphiques doivent effectuer une multiplication et une division de matrice sur plusieurs sommets en parallèle pour convertir ces coordonnées 3-D en coordonnées 2-D pour pouvoir afficher l'image sur l'écran. La carte graphique doit aussi effectuer plusieurs calculs à chaque pixel en parallèle pour déterminer la couleur de chaque pixel. Dans les deux cas, les calculs sont assez simples et n'impliquent pas beaucoup de branchement par rapport à la charge de travail informatique qu'un CPU rencontre habituellement.

Les réseaux de neurones bénéficient également des mêmes caractéristiques de performance. Les réseaux neuronaux impliquent généralement de nombreux paramètres, de valeurs d'activation

et de gradient, dont chacun doit être complètement mis à jour pendant chaque étape de l'entraînement. Ces quantités de paramètres sont suffisamment grandes pour tomber à l'extérieur du cache d'un ordinateur de bureau traditionnel, de sorte que la bande passante de mémoire du système devient souvent le facteur limitant de l'utilisation des CPU dans l'entraînement de réseaux de neurones. Les GPU offrent un avantage attractif par rapport aux CPU en raison de leur grande bande passante de mémoire. Ainsi l'utilisation des GPUs dans l'entraînement des réseaux de neurones est devenu la norme.

7.6 Résultats de l'extraction des mots clés

Table 6 – Mots clés extraits des commentaires positifs selon le niveau de satisfaction

Niveau de satisfaction	1	2	3	4	5
Keywords extraits	['conseil', 'souscrir', 'cas', 'problem', 'tres', 'agent', 'courti', 'servic', 'répar', 'correct', 'sinistr', 'client', 'condit', 'respect', 'garant', 'véhicul', 'autr', 'assureur', 'aucun', 'cher', 'pay', 'voitur', 'tarif', 'agenc', 'rien', 'contrat', 'prix', 'avantag']	['garant', 'réseau', 'bon', 'divers', 'agent', 'servic', 'assur', 'local', 'cas', 'réactiv']	['garant', 'conseiller', 'négoci', 'rapid', 'servic', 'bon', 'agenc', 'franc', 'écout', 'tarif', 'rembours', 'conseil', 'bien', 'disponibil', 'relationnel', 'réactiv', 'assist', 'contact', 'problem', 'accident', 'client', 'prix', 'départ', 'cher', 'qualit', 'interlocuteur', 'disponibl', 'intervent', 'général', 'fac', 'franchis', 'courti', 'habit', 'voitur']	['assur', 'correct', 'écout', 'sinistr', 'servic', 'agent', 'le', 'rapid', 'client', 'toujour', 'bon', 'intervent', 'tres', 'compétent', 'mem', 'rembours', 'garant', 'possibil', 'tarif', 'cas', 'disponibl', 'agenc', 'accueil', 'prix', 'rapport', 'qualit', 'assureur', 'contact', 'direct', 'conseiller']	['tres', 'bon', 'rapid', 'rembours']

Références

[1] Yoshua Bengio. A connectionist approach to speech recognition. *International journal of pattern recognition and artificial intelligence*, 7(04) :647–667, 1993.

[2] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb) :1137–1155, 2003.

[3] Hervé Bourlard and Nelson Morgan. Connectionist speech recognition-a hybrid approach. Technical report, KLUWER ACADEMIC PUBLISHERS, 1994.

[4] G. Casiez. Cours analyse en composantes connexes.

[5] Muller S. Charron C. « la réputation », mémoire réalisé dans le cadre du séminaire gestion des risques et environnement, master 2 innovation et management des technologies, université paris 1 panthéon sorbonne. 2011.

[6] Mohamed Cheriet, Nawwaf Kharma, Cheng-Lin Liu, and Ching Suen. *Character recognition systems : a guide for students and practitioners*. John Wiley & Sons, 2007.

[7] Luciana Dalla Valle, Dean Fantazzini, and Paolo Giudici. Copulae and operational risks.

[8] Xavier Dupré. *Contributions à la reconnaissance de l'écriture cursive à l'aide des modèles de Markov cachès*. PhD thesis, Paris 5, 2004.

[9] IL Elmhurst. Optical character recognition and the years ahead, 1969.

[10] Christophe Giraud. Analyse en composantes principales. `http://www.cmap.polytechnique.fr/~giraud/MSV/ACP.pdf`.

[11] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. Book in preparation for MIT Press, 2016.

[12] Alex Graves et al. *Supervised sequence labelling with recurrent neural networks*, volume 385. Springer.

[13] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification : labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006.

[14] Alex Graves and Jürgen Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. In *Advances in neural information processing systems*, pages 545–552, 2009.

[15] Emmanuèle Grosicki and Haikal El Abed. Icdar 2009 handwriting recognition competition. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*, pages 1398–1402. IEEE, 2009.

[16] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786) :504–507, 2006.

[17] Michael Irwin Jordan. *Learning in graphical models*, volume 89. Springer Science & Business Media, 1998.

[18] F. Kleber, S. Fiel, M. Diem, and R. Sablatnig. Cvl-database : An off-line database for writer retrieval, writer identification and word spotting. In *2013 12th International Conference on Document Analysis and Recognition*, pages 560–564, Aug 2013.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[20] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11) :2278–2324, 1998.

[21] Marcus Liwicki, Alex Graves, Horst Bunke, and Jürgen Schmidhuber. A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. In *Proc. 9th Int. Conf. on Document Analysis and Recognition*, volume 1, pages 367–371, 2007.

[22] Stéphane Mallat. Understanding deep convolutional networks. *Phil. Trans. R. Soc. A*, 374(2065) :20150203, 2016.

[23] U-V Marti and Horst Bunke. The iam-database : an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5(1) :39–46, 2002.

[24] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv :1301.3781*, 2013.

[25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[26] Boistel P. Le management de la réputation chez sernam : application du modèle ips. 2007.

[27] Vu Pham, Théodore Bluche, Christopher Kermorvant, and Jérôme Louradour. Dropout improves recurrent neural networks for handwriting recognition. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 285–290. IEEE, 2014.

[28] Martin F Porter. An algorithm for suffix stripping. *Program*, 14(3) :130–137, 1980.

[29] Princeton. Auerbach on optical character recognition, 1971.

[30] H. F Schantz. *The history of OCR, optical character recognition.* Manchester Center,Vt. : Recognition Technologies Users Association, 1982.

[31] Ray Smith. An overview of the tesseract ocr engine. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 2, pages 629–633. IEEE, 2007.

[32] Jan Snyman. *Practical mathematical optimization : an introduction to basic optimization theory and classical and new gradient-based algorithms*, volume 97. Springer Science & Business Media, 2005.

[33] Florent Pratlong Sophie Gaultier-Gaillard. Le risque de réputation : le cas du secteur bancaire. 2011.

[34] Robert Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5 : An open multilingual graph of general knowledge. 2017.

[35] Rafael Grompone von Gioi, Jérémie Jakubowicz, Jean-Michel Morel, and Gregory Randall. Lsd : a line segment detector. *Image Processing On Line*, 2 :35–55, 2012.