Conception itérative et semi-supervisée d'assistants conversationnels par regroupement interactif des questions

Erwan Schild*,**, Gautier Durantin*,
Jean-Charles Lamirel**, Florian Miconi*

* Euro-Information Développements, Groupe Crédit-Mutuel
4 Rue Frédéric-Guillaume Raiffeisen 67000 Strasbourg,
prenoms.nom@e-i.com, https://www.e-i.com/
** LORIA, Université de Lorraine
615 Rue du Jardin-Botanique, 54506 Vandoeuvre-lès-Nancy,
prenoms.nom@loria.fr, https://www.loria.fr/

Résumé. La création d'un jeu de données pour l'entrainement d'un chatbot repose sur un a priori de connaissance du domaine. En conséquence, cette étape est le plus souvent manuelle, fastidieuse et soumise aux biais. Pour garantir l'efficacité et l'objectivité de l'annotation, nous proposons une méthodologie d'apprentissage actif par annotation de contraintes. Il s'agit d'une approche itérative, reposant sur un algorithme de *clustering* pour segmenter les données et tirant parti de la connaissance de l'annotateur pour guider le regroupement des questions en une structure d'intentions. Dans cet article, nous étudions les paramètres optimaux de modélisation pour réaliser une segmentation exploitable en un minimum d'annotations, et montrons que cette approche permet d'aboutir à une structure cohérente pour l'entrainement d'un assistant conversationnel.

1 Introduction

L'utilisation d'assistants conversationnels («*chatbots*») est de plus en plus courante, que ce soit pour l'automatisation de demandes ou l'accès à des bases documentaires complexes (Goasduff, 2019; Costello, 2019). Ces derniers sont très appréciés car ils permettent l'accès à l'information en langage naturel, tout en offrant un gain de performance, de temps et de disponibilité dans le traitement des demandes.

La plupart des environnements de conception d'assistants sont basés sur une approche symbolique (Hoyt et al., 2016; Bocklisch et al., 2017; Alexa Internet, 2018), reposant sur la définition formelle de l'ensemble des intentions exprimées dans les demandes, éventuellement augmentées par la présence d'entités [1]. Cet ensemble d'intentions est en général bien défini dans un cas d'usage considéré (exemples : assistant de réservation, assistance bancaire, ...) De plus, cette approche permet en général une implémentation rapide et un bon contrôle des réponses, ce qui explique sa popularité.

1. Par exemple, «*Joue-moi du jazz!*» peut être modélisé par l'intention "*jouer de la musique*" et l'entité "*jazz*".

Pour la mise en oeuvre de ces assistants, la reconnaissance des intentions peut se faire au moyen d'une classification supervisée (les classes représentent alors les intentions).

Cependant, la littérature scientifique offre peu de choix de techniques pour concevoir un jeu de données fiable nécessaire à l'entrainement d'un assistant. En effet, cette phase de conception et d'annotation résulte le plus souvent d'un travail manuel, et cette approche empirique possède plusieurs défauts importants :

— avant de commencer le travail d'annotation, il faut entreprendre la définition d'un modèle de catégorisation des données en intentions : cette tâche est longue et complexe car elle suppose d'avoir une bonne connaissance du périmètre ;
— la modélisation de l'agencement des intentions obtenue est souvent théorique et éloignée de l'agencement réel à cause des a priori de conception ;
— l'étiquetage des données en fonction du modèle établi est difficile à réaliser sans introduire de confusion : en effet, deux annotateurs peuvent avoir une interprétation différente de l'agencement des intentions, et ainsi attribuer une étiquette différente pour certaines données ambigües à annoter [2] ;
— une évolution du périmètre de l'assistant (exemple : ajout de nouvelles fonctionnalités) implique une redéfinition de l'ensemble des intentions : cette approche n'est donc pas robuste car toute mise à jour nécessite un surcoût équivalent au coût initial.

En conséquence, cette approche manuelle est coûteuse, d'une part financièrement (intervention d'experts du domaine), d'autre part en temps humain (temps de modélisation des intentions et d'annotation et données).

Pour assister l'humain dans cette tâche d'annotation, une solution possible est l'introduction d'initiatives de la machine. Cela peut être mis en oeuvre par l'utilisation d'une classification non supervisée (*clustering*), où l'algorithme regroupe les données en fonction de leurs similarités intrinsèques. Mais malgré ses capacités d'exploration des données, les résultats du *clustering* manquent souvent de fiabilité. Parmi ses limites connues, on compte sa sensibilité aux données bruitées, sa difficulté à manipuler des espaces à grandes dimensions et son incapacité à exploiter des modèles de distribution et des métriques adaptés au problème qu'il traite. En outre, un de ses inconvénients majeurs réside en son manque de pertinence métier dans ses résultats, car les algorithmes de *clustering* n'arrivent pas à extraire seul les connaissances spécifiques au périmètre qui sont présentes dans les données (Xu et Tian, 2015). Tous ces problèmes sont caractéristiques du domaine du traitement du langage naturel, ce qui réduit l'utilité du *clustering* dans ce domaine.

Pour pallier ces problèmes, il existe des approches prometteuses basées sur la combinaison entre l'apprentissage actif et le *clustering* sous contraintes : l'apprentissage actif (*active learning*) permet d'optimiser les interactions entre l'homme et la machine au cours du processus d'apprentissage, tandis que le *clustering* sous-contraintes permet quant à lui d'optimiser les résultats de la phase d'apprentissage non supervisée. Nous faisons référence en particulier à la notion de *clustering* interactif proposé par Gançarski et Wemmert (2007) et Lampert et al. (2019) pour la détection d'objets dans une image et qui suit ce modèle.

2. Cette situation est d'autant plus forte pour le traitement du langage naturel où l'attribution d'une intention à une requête donnée peut avoir un caractère subjectif. Par exemple : est-ce que la question « *Que faire si j'ai perdu ma carte bancaire ?* » relève de l'intention "*déclaration de perte*", de l'intention "*commande de carte*" ou bien des deux ?

Toutefois, dans le domaine du traitement du langage naturel, la littérature scientifique offre peu de recul sur les choix de mises en oeuvre d'une méthode de *clustering* interactif. En particulier, il reste à identifier les facteurs significatifs influençant la performance de cette approche.

Dans cet article, nous proposons une adaptation de la méthode du *clustering* interactif pour le traitement du langage naturel afin d'assister la création d'un assistant conversationnel. Cette adaptation est accompagnée d'une expérience d'initialisation de chatbot dans laquelle nous manipulons quatre paramètres [3] d'implémentation du *clustering* interactif, et nous mesurons leur effet sur l'effort d'annotation à fournir pour obtenir une structure de référence issue de données réelles. L'objectif de notre expérience est de : (1) vérifier la pertinence des implémentations du *clustering* interactif pour converger vers la vérité terrain établie, et (2) identifier les paramètres ayant une influence sur cette convergence ainsi que leur(s) valeur(s) optimale(s).

2 Contexte de l'approche

2.1 L'annotation par contraintes

Cette approche consiste à décrire la similarité entre les données par un ensemble de liens élémentaires plutôt que d'attribuer explicitement un label à chaque donnée. Dans la littérature, deux types de liens sont souvent exploités (Wagstaff et Cardie, 2000) : *MUST-LINK* (les données sont similaires) et *CANNOT-LINK* (les données ne sont pas similaires). Couplés avec un algorithme d'optimisation de contraintes, ces contraintes permettent de faire émerger naturellement la structure d'intentions. Parmi les avantages de cette méthode, on dénote une grande facilité et flexibilité à l'étiquetage des données car il n'est pas nécessaire de connaître précisément le périmètre des intentions. D'autre part, le travail investi peut être facilement réutilisé en cas de changement de périmètre puisque toute mise à jour peut être effectuée par l'ajout de nouvelles contraintes dans le problème d'optimisation.

2.2 Le clustering sous contraintes

Le *clustering* sous contraintes est une variante semi-supervisée des méthodes de clustering usuelles. Cette alternative consiste à influencer le fonctionnement du *clustering* en spécifiant des liens a priori entre les données, par des annotations ou par l'utilisation d'heuristiques, dans le but de corriger certaines de ses limites. On peut citer quelques exemples issus de la littérature (Lampert et al., 2018).

L'algorithme COP K-Means (Wagstaff et al., 2001). L'algorithme K-means (MacQueen, 1967) est une méthode de *clustering* usuelle qui repose sur la minimisation de l'inertie intra-classe. Dans la version sous contraintes, on s'assure en plus que l'affectation d'une donnée au cluster le plus proche ne viole aucune des contraintes imposées. La mise en oeuvre de cette version est simple, mais son exécution peut mener à des contradictions sans solution.

Le clustering hiérarchique sous contraintes (Davidson et Ravi, 2005). L'algorithme de *clustering* hiérarchique (Murtagh et Contreras, 2012) est une méthode itérative de fusion des

3. Le prétraitement, la vectorisation, la sélection des données à annoter, l'algorithme de *clustering* sous contraintes.

clusters les plus similaires. Cette stratégie de fusion peut être définie en s'appuyant sur plusieurs types de liens de similarité [4]. Dans la version sous contraintes, la stratégie de fusion vise à agglomérer en priorité les clusters ayant des liens *MUST-LINK* et à empêcher la fusion si deux clusters ont un lien *CANNOT-LINK*. Adapter cette méthode aux contraintes est trivial car cela revient simplement à modifier le calcul des liens de similarité entre clusters.

Le clustering spectral sous contraintes (Kamvar et al., 2003). Le *clustering* spectral (Ng et al., 2002) consiste à modéliser la matrice de similarité entre les données par ses vecteurs propres, et à regrouper ces derniers avec un algorithme de type K-Means. Cette approche permet de trouver des clusters ayant des formes complexes. Dans la version sous contraintes, on modifie la matrice de similarité en forçant sa valeur à 0 (respectivement à 1) si les deux données sont reliées par un lien *MUST-LINK* (respectivement *CANNOT-LINK*). L'adaptation nécessite donc peu d'effort, mais l'ajout d'une contrainte peut cependant entrainer dans certains cas un changement radical du fonctionnement de l'algorithme.

2.3 L'apprentissage actif

Le principe essentiel de l'apprentissage actif est de tirer le meilleur parti des connaissances de l'homme et des capacités de la machine. Dans ce processus itératif, l'humain corrige le résultat proposé par la machine, et la machine exploite ensuite ces corrections pour améliorer ses itérations ultérieures. Pour optimiser la pertinence des corrections apportées par l'humain, le processus d'apprentissage actif fait en général appel à un oracle qui est garant de la stratégie d'amélioration avec pour but d'obtenir un maximum d'efficacité. Diverses stratégies d'amélioration peuvent être proposées via cet oracle, comme par exemple la vérification d'une incertitude ou d'une certitude du système, la correction d'une erreur estimée du résultat obtenu ou encore l'exploration d'une partie inconnue des données (Settles, 2010). L'intérêt de suivre cette stratégie est de ne plus se limiter uniquement à un processus manuel (annotation) ou aveugle (*clustering*), mais de demander à l'humain d'introduire la dose suffisante de connaissance pour que la machine puisse bénéficier d'une efficacité maximale.

3 Clustering interactif

3.1 Principe général

La méthode que nous proposons repose sur l'alternance successive entre :

1. une phase d'annotation de contraintes (cf. section 2.1) : un oracle suggère des liens à annoter entre les données, permettant ainsi à l'annotateur de corriger le résultat du *clustering* de l'itération précédente ;
2. une phase de clustering de contraintes (cf. section 2.2) : le *clustering* exploite ces nouvelles annotations pour suggérer un nouveau partitionnement plus pertinent.

4. Lien simple (*single*) : fusion des deux clusters ayant les frontières les plus proches ; Lien complet (*complete*) : fusion des deux clusters ayant les frontières opposées les plus proches ; Lien moyen (*average*) : fusion des deux clusters ayant les centres de clusters les plus proches ; Lien de *Ward* : fusion des deux clusters donnant lieu à un nouveau cluster le plus compact possible.

L'objectif recherché par la combinaison de ces deux phases est la création d'un cercle vertueux pour améliorer itérativement la qualité du jeu de données obtenu.

3.2 Première itération - Initialisation

Nous disposons d'un jeu de questions (brut, non annoté) et de la définition du périmètre de l'assistant conversationnel à créer (i.e. on sait si l'assistant pourra répondre ou non à une question donnée). Dans cette étape, nous réalisons un premier *clustering* simple (non contraint) afin d'obtenir un regroupement initial des données.

3.3 Itération *N* - Phase d'annotation

Nous possédons un ensemble de clusters respectant les contraintes déjà à disposition. L'objectif est de soumettre des contraintes supplémentaires pour corriger le *clustering* courant.

Afin d'améliorer efficacement le *clustering* et limiter les biais introduits par l'annotation, un oracle (cf. section 2.3) est utilisé pour sélectionner les couples de questions à annoter. Ici, nous proposerons quatre stratégies pour cet oracle :

1. sélection des questions *de manière purement aléatoire* (stratégie témoin) ;
2. sélection de questions *aléatoires issues d'un même cluster* (stratégie visant à vérifier aléatoirement l'homogénéité d'un cluster) ;
3. sélection des questions *les plus éloignées issues d'un même cluster* (stratégie visant à s'assurer qu'un cluster ne phagocyte pas les intentions à ses frontières) ;
4. sélection des questions *les plus proches issues de deux clusters différents* (stratégie visant à valider la position des frontières entre les clusters).

Pour rester objectif dans l'attribution d'une contrainte, l'annotation ne se fait pas sur la similarité des questions, mais sur la similarité de la réponse apportée à ces dernières. En pratique, cela revient à s'interroger de la manière suivante : "*La réponse à apporter à ces questions est-elle la même ?*". Si c'est le cas, alors la contrainte est de type *MUST-LINK*, sinon elle est de type *CANNOT-LINK*.

3.4 Itération *N* - Phase de *clustering* sous contraintes

Après la phase d'annotation, nous disposons d'un lot de contraintes plus étoffé qu'à l'itération précédente. L'objectif est de proposer un nouveau partitionnement des données respectant toutes les contraintes disponibles. Plusieurs algorithmes de *clustering* sous contraintes peuvent être employés pour réaliser cette tâche. Le choix du l'algorithme peut être déterminé en fonction du cas d'usage et de la topologie des clusters à identifier.

4 Expérience

4.1 Jeu de données

Comme vérité terrain de cette expérience, nous utilisons un jeu de données inspiré d'un assistant conversationnel du domaine bancaire et dont le périmètre couvre les actions relatives

à la gestion des cartes de paiements. Ce jeu de données comporte 6 intentions de 25 questions chacune et la taille totale du vocabulaire est de 215 tokens [5]. Les intentions présentes ont été créés par un expert du domaine en amont de l'expérience, de telle sorte que toutes les questions présentes dans une même intention possèdent la même réponse. Le tableau 1 donne un bref aperçu du jeu de données.

INTENTION	EXEMPLE
carte avalée	«*Ma carte a été avalée par un distributeur !*»
commande de carte	«*Est-ce que je peux avoir une carte sans frais ?*»
consultation du solde	«*Comment savoir si je suis à découvert ?*»
déblocage de carte	«*Débloquer sa carte après trois mauvais codes*»
gestion du plafond	«*Augmenter mon plafond de paiement à l'étranger*»
gestion du sans contact	«*Activer l'option NFC sur ma Mastercard*»

TAB. 1 – *Exemple de questions pour chaque intention du jeu de données utilisé.*

4.2 Protocole expérimental

Pour valider nos hypothèses sur le *clustering* interactif, nous essayons de créer un *clustering* pertinent en appliquant des itérations successives de la méthodologie décrite en section 3 sur les données non annotées. Nous supposons que le nombre de clusters est connu et qu'il correspond au nombre d'intentions établies. La taille de chaque lot d'annotation est fixée à 25 couples de questions, et l'annotateur humain sera quant à lui représenté symboliquement par une annotation automatique des contraintes sur la base de la comparaison des labels de vérité terrain. Comme l'annotation se base sur la vérité terrain, il n'est pas possible d'obtenir de contradictions dans notre graphe de contraintes. Toutefois, ce problème sera étudié ultérieurement afin de valider la méthode pour l'application en situation réelle.

Dans cette expérience, nous étudions les paramètres d'implémentation [6] suivants :

1. deux types de **prétraitements** : un prétraitement simple (suppression des accents et de la ponctuation) et un prétraitement avancé (ajout d'une lemmatisation [7] et de la suppression des mots vides) ;
2. deux types de **vectorisation** : des vecteurs basés sur la fréquence des termes (*TF-IDF* [8]) et des vecteurs à base d'*embeddings* [9] ;
3. trois implémentations de **clustering sous contraintes** [10] basées sur les descriptions de la section 2.2 (la méthode hiérarchique utilisant quatre types de distances différentes) ;
4. quatre implémentations de **sélection de contraintes** issues de la section 3.3.

Il y a au total 96 combinaisons de paramètres pour 4 facteurs d'analyse, chaque combinaison faisant l'objet de 20 observations, soit 1920 observations en tout.

5. Tokenisation réalisée avec la librairie *spacy* (Honnibal et Montani, 2017).
6. Implémentation réalisée en *Python 3.6* (Van Rossum et Drake, 2009).
7. Lemmatisation avec la librairie *spacy* (Honnibal et Montani, 2017).
8. Vectorisation réalisée avec la librairie *sklearn* (Pedregosa et al., 2011).
9. Vecteurs entrainés sur un corpus bancaire à l'aide de la librairie *FastText* (Bojanowski et al., 2016).
10. Le *clustering* initial repose sur le même algorithme, mais exécuté sans contrainte sur les données.

Pour quantifier la performance du *clustering* sous contraintes, nous calculons pour chaque itération les mesures de qualités usuelles (homogénéité, complétude et V-mesure) [11] entre le résultat du *clustering* et la vérité terrain. Ensuite, nous nous intéressons au nombre moyen d'annotations nécessaires pour atteindre un seuil de performance (en terme de V-mesure) au cours des itérations [12]. Dans cette étude, nous nous intéressons aux seuils suivant :

— un seuil de 80% de V-mesure, représentant une annotation partielle et correspondant à un *clustering* conforme à 80% à la vérité terrain ;
— un seuil de 99% de V-mesure, représentant une annotation exhaustive ;
— l'annotation d'un nombre suffisamment de contraintes pour pouvoir déduire la vérité terrain par transitivité, représentant ainsi une annotation complète.

L'analyse statistique de l'effet des paramètres de la modélisation sur le nombre d'annotations requis est réalisée à l'aide du logiciel R (R Core Team, 2017), par une ANOVA à mesures répétées. Le Tukey HSD est utilisé pour les comparaisons post-hoc.

5 Résultats

On constate que les observations convergent vers la vérité terrain au cours des itérations :

— à l'initialisation, pour un *clustering* sans contraintes, la V-mesure moyenne obtenue est de 30% ($min = 9\%$, $max = 60\%$, $\sigma = 15\%$) ;
— pour une annotation partielle à 80% de V-mesure, il faut en moyenne 16.7 itérations ($min = 4$, $max = 61$, $\sigma = 8.1$) soit 418 annotations ; le minimum d'annotation est de 101 dont 51 contraintes de type *MUST-LINK* ;
— pour une annotation exhaustive à 99% de V-mesure, il faut en moyenne 22.0 itérations ($min = 9$, $max = 66$, $\sigma = 8.0$), soit 550 annotations ; le minimum d'annotation est de 255 dont 108 contraintes de type *MUST-LINK* ;
— pour fournir une annotation complète de tous les liens possibles, il faut en moyenne 23.8 itérations ($min = 11$, $max = 74$, $\sigma = 8.8$), soit 595 annotations.

Les tableaux 2 et 3 retranscrivent les influences de chacun des paramètres sur le nombre d'itérations nécessaires à la convergence vers la structure de référence. Les analyses de variance mettent en relief l'effet significatif sur cette convergence à 99% de V-mesure de la vectorisation ($\eta^2 = 0.885$, $p < 10^{-3}$), de l'algorithme de clustering utilisé ($\eta^2 = 0.977$, $p < 10^{-3}$) et de la méthode de sélection des contraintes ($\eta^2 = 0.996$, $p < 10^{-3}$). L'analyse post-hoc de ces effets indique que le meilleur paramétrage moyen repose sur la vectorisation *TF-IDF* , le *clustering* hiérarchique (lien moyen) et la sélection de données semi-aléatoire ou basée sur les voisins les plus proches sur la frontière des clusters. La moyenne du nombre d'itération requis pour ce paramétrage est de 12.9 ($\sigma = 2.6$).

Pour la convergence à 80% de V-mesure, l'ANOVA montre un effet significatif des quatre paramètres considérés. L'analyse post-hoc indique que le meilleur paramétrage moyen serait composé d'un prétraitement simple, de la vectorisation *FastText*, du *clustering* hiérarchique (lien simple) et d'une sélection de données basée sur les voisins les plus proches sur la frontière des clusters. Ainsi, la moyenne du nombre d'itération requis est de 10.2 ($\sigma = 0.9$).

11. Mesures réalisées à l'aide du module *metrics* de la librairie *scikit-learn* (Pedregosa et al., 2011).
12. L'effort humain et le temps de calcul ne sont pas estimés ici, mais pourront être traité ultérieurement sur un cas pratique.

Facteur	Niveau	STATISTIQUES DESCRIPTIVES Moyenne	SE	MESURE DES TAILLES D'EFFET η^2	p-valeur
prétraitement	**simple**	16.5 (1)	0.05	0.347	$4.99e^{-3}$ **
	avancé	16.8 (2)			
vectorisation	**fasttext**	16.3 (1)	0.05	0.679	$4.34e^{-6}$ ***
	tfidf	17.0 (2)			
algorithme de *clustering*	**hier. simple**	11.6 (1)	0.09	0.993	$< 2e^{-16}$ ***
	hier. moyen	12.3 (2)			
	cop-kmeans	15.0 (3)			
	hier. complet	18.4 (4)			
	hier. ward	19.1 (5)			
	spectral	23.5 (6)			
sélection de contraintes	**plus proches**	9.4 (1)	0.07	0.997	$< 2e^{-16}$ ***
	semi-aléatoire	14.1 (2)			
	aléatoire	20.1 (3)			
	plus éloignées	22.1 (4)			

TAB. 2 – *ANOVA du nombre d'itérations nécessaires pour l'obtention de 80% de V-mesure. Les (∗) dénotent le niveau de significativité ($\alpha = 0.05$). Pour les effets significatifs, les chiffres précisés entre parenthèses dans la colonne Moyenne indiquent le classement des niveaux selon les analyses post-hoc.*

Facteur	Niveau	STATISTIQUES DESCRIPTIVES Moyenne	SE	MESURE DES TAILLES D'EFFET η^2	p-valeur
prétraitement	simple	21.9	0.06	0.162	0.0703 ·
	avancé	22.1			
vectorisation	**tfidf**	21.4 (1)	0.06	0.885	$2.29e^{-10}$ ***
	fasttext	22.7 (2)			
algorithme de *clustering*	**hier. moyen**	19.6 (1)	0.11	0.977	$< 2e^{-16}$ ***
	cop-kmeans	20.1 (2)			
	hier. simple	20.6 (3)			
	hier. complet	22.3 (4)			
	hier. ward	22.8 (5)			
	spectral	26.9 (6)			
sélection des contraintes	**plus proches**	17.5 (1)	0.09	0.996	$< 2e^{-16}$ ***
	semi-aléatoire	17.5 (1)			
	aléatoire	26.5 (3)			
	plus éloignées	26.7 (3)			

TAB. 3 – *ANOVA du nombre d'itérations nécessaires pour l'obtention de 99% de V-mesure. Les (∗) dénotent le niveau de significativité ($\alpha = 0.05$). Pour les effets significatifs, les chiffres précisés entre parenthèses dans la colonne Moyenne indiquent le classement des niveaux selon les analyses post-hoc.*

La figure 1 représente une comparaison de l'évolution des performances moyennes pour chacun des meilleurs paramétrages moyens pour 80% et 99% de V-mesure.

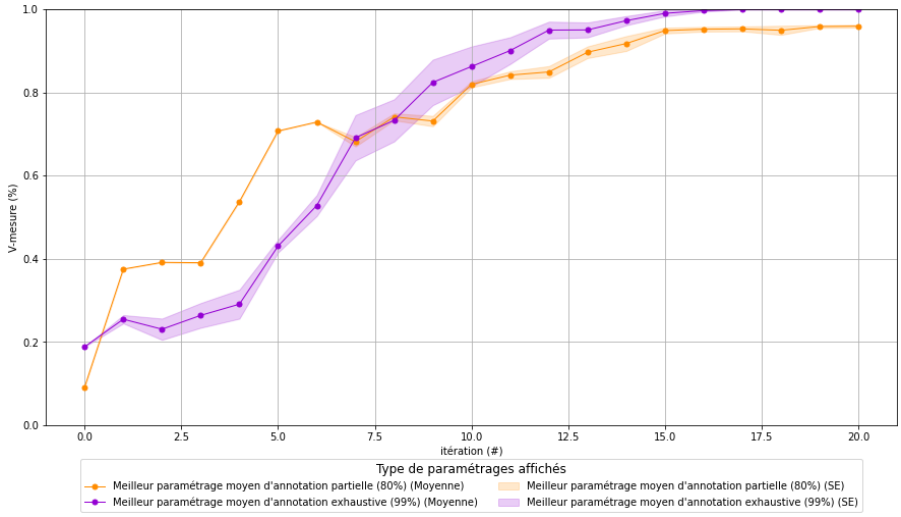

FIG. 1 – *Comparaison de l'évolution de la V-mesure moyenne au cours des itérations entre les meilleurs paramétrages pour les indicateurs d'annotation partielle (80%) et d'annotation exhaustive (99%). Les barres d'erreurs représentent l'erreur standard de la moyenne.*

6 Discussions et conclusions

Dans cette expérience, nous proposions d'étudier une implémentation du *clustering* interactif pour l'annotation de données textuelles.

Notre première hypothèse concernait l'analyse de la pertinence de cette implémentation. Nous confirmons tout d'abord l'insuffisance du *clustering* utilisé seul pour fournir une structure d'intentions utilisable. En effet, la première itération de l'expérience ne mène en moyenne qu'à une V-mesure de 30%. Le rôle de l'humain est donc bien à conserver. Ensuite, avec à la mise en place du *clustering* interactif, toutes les observations convergent bien, au cours des itérations, vers la structure de référence (avec une marge d'erreur de 1%). Une annotation exhaustive mène bien à la vérité terrain établie. Notre expérience montre donc que la méthode permet de créer itérativement la structure d'intentions à l'aide d'annotation par contraintes.

Ce constat implique plusieurs gains pour la phase d'annotation. D'une part, il n'est plus nécessaire de définir au préalable la modélisation en intentions des données pour les annoter : celle-ci émerge naturellement de l'annotation. Ceci est un gain de temps pour l'initialisation d'un assistant conversationnel. D'autre part, l'annotation est à caractère binaire, ce qui n'impose plus à l'annotateur d'avoir une connaissance globale de la structure d'intentions. Cette annotation est par conséquent simplifiée. Enfin, la tâche est ici partagée entre l'humain et la machine : l'humain annote ce que lui suggère la machine via l'oracle, et la machine trie les données suivant les contraintes que lui donne l'annotateur.

Toutefois, il reste un prérequis essentiel à cette méthode pour que l'étiquetage des données reste fiable : l'annotateur doit rester un expert du domaine capable de distinguer si deux questions possèdent ou non la même réponse.

Notre seconde hypothèse s'intéressait à la présence de facteurs influençant la rapidité de la convergence du *clustering* interactif ainsi qu'à leur valeur optimales. En accord avec les analyses ANOVA (cf. tableaux 2 et 3), nous avons montré que le choix de la vectorisation, le choix de la sélection des données pour l'annotation et le choix de l'algorithme de *clustering* sous contraintes sont les trois facteurs intervenant toujours de manière significative.

À propos de la vectorisation, sa pertinence dépend de l'exhaustivité de l'annotation voulue. Pour une annotation partielle, les vecteurs à base d'*embeddings* semblent plus efficaces (probablement grâce à l'apport de connaissance qu'ils offrent), mais ce gain s'estompe pour réaliser rapidement une annotation exhaustive (probablement car la connaissance a priori des *embeddings* ne correspond pas à la vérité terrain sur certaines questions spécifiques).

En ce qui concerne la sélection des données pour l'annotation, la plus pertinente est la sélection des plus proches voisins issus de deux clusters différents (i.e. sur une frontière entre clusters). Les gains de performances observés peuvent être expliqués par le plus grand nombre de liens *MUST-LINK* annotés, ce qui permet de corriger efficacement la position des frontières. À l'inverse, la sélection des voisins les plus éloignés dans un même cluster s'est révélée très peu pertinente, probablement parce qu'elle propose plus de contraintes de type *CANNOT-LINK* qui donnent peu d'information sur la manière de corriger les clusters.

Au sujet de l'algorithme de *clustering* sous contraintes, le *clustering* hiérarchique (plus particulièrement les implémentations des liens simple et moyen) s'est révélé le plus prometteur. Cela peut s'expliquer par sa relative stabilité lors de l'ajout d'une nouvelle contrainte, ce qui se traduit par une performance globalement croissante au cours des annotations. À l'inverse, le *clustering* spectral peut voir son résultat radicalement changer suite à l'ajout d'une seule contrainte, ce qui le rend relativement instable et donc moins performant.

Quant aux prétraitements, ils n'ont qu'une faible importance malgré leur significativité (gain inférieur à une itération).

De manière générale, le meilleur paramétrage moyen significatif dépend du choix de la sélection des contraintes (sélection des voisins les plus proches issus de deux clusters différents) et de l'algorithme de *clustering* sous contraintes (algorithme hiérarchique, avec lien simple pour une annotation partielle et avec lien moyen pour une annotation exhaustive).

En conclusion, le paramétrage que nous proposons permet, au regard de l'expérience réalisée, d'obtenir une première implémentation fonctionnelle et viable de la méthode du *clustering interactif* pour annoter des données textuelles. Cela montre bien qu'il est possible d'introduire des initiatives de la machine, telles que le *clustering* interactif, pour accélérer la création d'un assistant conversationnel.

Il reste cependant plusieurs pistes à explorer et à valider :

— dans cette étude, la taille du jeu de données est volontairement faible pour faciliter la découverte d'un paramétrage supposé optimal : la prochaine étape sera d'implémenter un assistant de bout en bout pour appliquer la méthode dans un cas pratique ;
— d'autres facteurs ou valeurs de paramétrages peuvent être étudiés pour parfaire l'implémentation de la méthodologie proposée : par exemple, la taille du lot d'annotation peut avoir un impact sur la convergence, et l'utilisation d'un *clustering* collaboratif permettrait traiter le choix du le nombre de clusters optimal en le déduisant itérativement ;

— les contradictions entre les annotations peuvent être révélatrices d'ambiguïtés, et ces informations non exploitées ici seront à traiter pour prévenir des dérives potentielles de l'annotation ;
— si l'obtention d'une structure identique à celle de référence semble complexe à mettre en oeuvre car elle requiert un nombre d'annotations supérieur au nombre de données (au minimum 255 dans notre expérience), l'obtention d'un haut niveau de conformité (80% de V-mesure) avec le jeu de référence est possible moyennant un nombre d'annotations inférieur (au minimum 101 dans notre expérience). Compte tenu de la subjectivité présente dans le jeu de données de références, il serait légitime de se demander si ce jeu de données intermédiaire permettrait l'implémentation d'un assistant performant.

Toutefois, si l'exploration de ces pistes permettait de clarifier d'avantage la manière de mettre en oeuvre le *clustering* interactif, notre approche confirmerait encore plus clairement un moyen de remettre en cause l'annotation par labels (longue, coûteuse et basée sur la connaissance à priori des intentions) par une annotation semi-supervisée implémentée par le *clustering* interactif (plus rapide, plus simple, basée sur la réponse). Cette technique serait alors un réel soutien pour l'assistance à la création de nouveaux assistants conversationnels.

Références

Alexa Internet (2018). Keyword Research, Competitor Analysis, and Website Ranking : Alexa.

Bocklisch, T., J. Faulkner, N. Pawlowski, et A. Nichol (2017). Rasa: Open Source Language Understanding and Dialogue Management. *arXiv:1712.05181*.

Bojanowski, P., E. Grave, A. Joulin, et T. Mikolov (2016). Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606*.

Costello, K. (2019). Gartner Top Technologies and Trends Driving the Digital Workplace. *Gartner, Inc*.

Davidson, I. et S. S. Ravi (2005). Agglomerative Hierarchical Clustering with Constraints : Theoretical and Empirical Results. *Springer, Berlin, Heidelberg 3721*, 12.

Gançarski, P. et C. Wemmert (2007). Collaborative multi-step mono-level multi-strategy classification. *Multimedia Tools and Applications 35*(1), 1–27.

Goasduff, L. (2019). Chatbots Will Appeal to Modern Workers. *Gartner, Inc*.

Honnibal, M. et I. Montani (2017). spaCy 2 : Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.

Hoyt, R. E., D. Snider, C. Thompson, et S. Mantravadi (2016). IBM Watson Analytics : Automating Visualization, Descriptive, and Predictive Statistics. *JMIR Public Health Surveill 2*(2).

Kamvar, S. D., D. Klein, et C. D. Manning (2003). Spectral Learning. *Proceedings of the international joint conference on artificial intelligence*, 561–566.

Lampert, T., T.-B.-H. Dao, B. Lafabregue, N. Serrette, G. Forestier, B. Cremilleux, C. Vrain, et P. Gancarski (2018). Constrained distance based clustering for time-series : a comparative and experimental study. *Data Mining and Knowledge Discovery 32*(6), 1663–1707.

Lampert, T., B. Lafabregue, et P. Gançarski (2019). Constrained Distance based K-Means Clustering for Satellite Image Time-Series. In *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, pp. 2419–2422. IEEE.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability 1*(14), 281–297.

Murtagh, F. et P. Contreras (2012). Algorithms for hierarchical clustering : An overview. *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery 2*, 86–97.

Ng, A. Y., M. I. Jordan, et Y. Weiss (2002). On Spectral Clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, et Z. Ghahramani (Eds.), *Advances in Neural Information Processing Systems 14*. MIT Press.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, et E. Duchesnay (2011). Scikit-learn : Machine Learning in Python. *Journal of Machine Learning Research 12*, 2825–2830.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.

Settles, B. (2010). Active Learning Literature survey.

Van Rossum, G. et F. L. Drake (2009). *Python 3 Reference Manual* (CreateSpace ed.). Scotts Valley, CA.

Wagstaff, K. et C. Cardie (2000). Clustering with Instance-level Constraints. *Proceedings of the Seventeenth International Conference on Machine Learning*, 1103–1110.

Wagstaff, K., C. Cardie, S. Rogers, et S. Schroedl (2001). Constrained K-means Clustering with Background Knowledge. *International Conference on Machine Learning*.

Xu, D. et Y. Tian (2015). A Comprehensive Survey of Clustering Algorithms. *Annals of Data Science 2*, 165–193.

Summary

The design of a dataset needed to train a chatbot is most often the result of manual and tedious step. To guarantee the efficiency and objectivity of the annotation, we propose an active learning method based on constraints annotation. It's an iterative approach, relying on a clustering algorithm to segment data and using annotator knowledge to lead clustering from unlabeled question to relevant intents structure. In this paper, we study the optimal modeling parameters to get an exploitable dataset with a minimum of annotations, and show that this approach allows to make a coherent structure for the training of a chatbot.