évaluation de métriques d'estimation de l'accord inter-annotateur

Dany Brégeon

L3 informatique

avril-juin 2018

Remerciements

Je tiens à remercier Jean-Yves Antoine, Anaïs Lefeuvre et Jeanne Villaneau pour m'avoir donné l'opportunité de faire ce stage, de m'avoir guidé et soutenu tout au long de celui-ci. Un grand merci pour leur confiance, leurs conseils, pour le temps passé ensemble. Je les remercie pour toujours avoir été à l'écoute, que ce soit en réunion ou par mail. Je remercie Anaïs également pour m'avoir emmené à Blois, ainsi que Jean-Yves pour les repas.

Table des matières

Chapitre 1

Introduction

Le machine learning (techniques statistiques d'apprentissage automatique) et le deep learning (réseaux de neurones artificiels) sont de plus en plus utilisés partout, et utilisent souvent des données qui ont été obtenues par annotation manuelle réalisées par la foule ou par des experts humains pour leur entraînement.

La qualité de ces données est évaluée par des métriques qui estiment l'accord entre les annotateurs. En effet, plus une foule d'annotateur s'accorde sur une annotation, plus celle-ci a des chances a priori d'être fiable.

Le Laboratoire d'Informatique de l'Université de Tours (LI) propose ce stage financé par l'équipe BDTLN du laboratoire, en collaboration avec le laboratoire LIFO de l'Université d'Orléans et l'IRISA Lorient (INRIA). Ce stage fait suite à une collaboration déjà initiée par le passé sur l'étude de la validité des métriques d'accord inter-annotateur [1], et porte sur l'évaluation de ces métriques : il est question de faire une étude expérimentale de la validité de mesures d'accord inter-annotateur par génération aléatoire de données bruitées, et d'observer le comportement de ces métriques.

Dans le cadre de ma licence d'informatique, j'ai souhaité réaliser mon stage dans la recherche, ayant déjà eu une expérience en entreprise par le passé, et voulant voir si la recherche me plaisait. Le sujet du stage a tout d'abord éveillé ma curiosité, et après avoir lu l'article scientifique sur l'étude de la validité des métriques d'accord inter-annotateur, m'a très fortement intéressé. De plus, l'aspect un peu mathématiques du sujet m'a également plu, aimant beaucoup les mathématiques.

Dans un premier temps nous nous intéresserons à la compréhension du sujet, par l'explication de vocabulaires utilisés et du fonctionnement des métriques d'accord inter-annotateur. Puis nous étudierons le travail que j'ai effectué durant ce stage, avant de finir par un bilan de celui-ci.

Chapitre 2

Contexte du stage

2.1 Définitions

Il est nécessaire pour la bonne compréhension du sujet de bien comprendre le sens des mots utilisés. En effet, il y aura une utilisation récurrente de mots tels que annotateur, annotation, classe, observable, corpus... On va donc commencé par voir leur signification :

Annotation : une méta-donnée sur un texte.

Classe : une catégorie d'annotation.

Annotateur : personne qui annote un texte, ici l'annotation doit être une des classes possible.

Corpus : ensemble fini de textes/de phrases choisi comme base d'une étude. Dans ce stage, les expériences ont été effectué sur 3 corpus différents :

-émotion, pour lequel les annotateurs devait évaluer leurs émotions sur des phrases issues de contes de fée parmi 5 catégories possible allant de très négative à très positive.

-opinion, où les observables sont des critique de films, et où les catégories sont les mêmes que pour émotion.

-coréférence, où il est question d'annoter des relations sur des discussions ou des discours entre coréférence direct, indirect, anaphore, anaphore associative, et anaphore associative pronominal.

Un quatrième corpus s'est également ajouté au cours du stage : le corpus similarité, dans lequel les annotateurs doivent estimer la ressemblance entre 2 phrases sur une échelle de 0 à 4.

2.2 Les métriques

Historiquement, le pourcentage d'accord (nombre d'accord / nombre total de potentiel accord) était utilisé pour déterminer l'accord inter-annotateur. Cependant, l'accord qui arrive par chance, dû aux suppositions des annotateurs, est une possibilité (de la même manière qu'avoir une bonne réponse à un qcm peut être dû à la chance). Le pourcentage d'accord ne prenant pas en compte cet accord par la chance, il surestime donc le niveau d'accord : avoir un bon accord brut n'a pas la même signification suivant que la chance d'arriver à cet accord était grande ou faible (par exemple un faible nombre de classe favorise l'accord brut), il est donc nécessaire de prendre en compte l'accord qui arrive par chance.

Les métriques étudiées estime donc l'accord entre plusieurs annotateurs en prenant en compte l'accord qui arrive par chance. La valeur de ces métriques varie entre 0 et 1, 0 étant aucun accord (ou plutôt un accord inférieur ou égal à celui qui serait arrivé par chance) et 1 un accord parfait. Ce qui doit être estimé est donc la proportion de l'accord observé au-delà de l'accord qui arrive par chance :

Mesure = $\frac{A_o - A_e}{1 - A_e}$

Avec A_o qui correspond à l'accord observé entre les annotateurs et A_e qui correspond à l'estimation d'un accord aléatoire. Ces métriques diffère par leur manière d'estimer l'accord qui arrive par chance.

2.2.1 Kappa de Cohen

La métrique la plus utilisé actuellement est le kappa κ de Cohen.

Pour comprendre son fonctionnement, prenons un exemple : Supposons deux évaluateurs (A et B) qui sont en charges de définir dans un groupe de 50 étudiants ceux qui seront reçu ou non. Chacun d'eux contrôle la copie de chaque étudiant et la note comme reçu ou non reçu (OUI ou NON). Le tableau ci dessous relate les résultats :

		A	
		OUI	NON
B	OUI	20	5
	NON	10	15

Dans cet exemple, 20 copies ont été notées oui par les deux enseignants, 15 copies ont été notées non par les deux enseignants. Au total, l'enseignant A a noté 30 copies oui et 20 copies non, et l'enseignant B a noté 25 copies oui et 25 copies non.

La première étape est de calculer A_o : les enseignants sont d'accord sur 35 des 50 cas (20 où les deux ont mis oui et 15 où les deux ont mis non). Donc

$A_o = \frac{nbAccord}{total} = \frac{20+15}{50} = 0.7$

La deuxième étape est de calculer A_e. Pour cela, il faut calculer la probabilité attendu que les deux correcteurs notent oui, et la probabilité attendu que les deux correcteurs notent non.

L'enseignant A a noté oui à 30 copies sur 50, soit 60%. L'enseignant B a noté oui à 25 copies sur 50, soit 50%. La probabilité que les deux enseignants notent oui aléatoirement est donc de

$0.6 * 0.5 = 0.3$

De même avec les cas où les enseignant ont noté non, on obtient

$0.4 * 0.5 = 0.2$

La probabilité que les enseignants soient en accord est donc de

$A_e = 0.3 + 0.2 = 0.5$

La formule du κ nous donne donc

$\kappa = \frac{A_o - A_e}{1 - A_e} = \frac{0.7 - 0.5}{1 - 0.5} = 0.4$

Landis et Koch ont proposé la table suivante pour interpréter le κ de Cohen. Il s'agit d'ordres de grandeurs qui ne font pas consensus dans la communauté scientifique, notamment parce que le nombre de catégories influe sur l'estimation obtenue : moins il y a de catégories, plus le κ est élevé.

κ	Interprétation
0	Accord équivalent à la chance
0.01 - 0.20	Accord très faible
0.21 - 0.40	Accord faible
0.41 - 0.60	Accord modéré
0.61 - 0.80	Accord fort
0.81 - 1.00	Accord presque parfait

2.2.2 Alpha de Krippendorff

Contrairement au κ de Cohen et à la plupart des autres métriques, l'alpha de Krippendorff est calculé à partir du désaccord entre les annotateurs et non plus de l'accord :

$\alpha = \frac{D_e - D_o}{D_e}$

Une autre particularité de cette métrique est qu'elle peut également prendre en compte une certaine distance entre les classes, et fonctionne pour n'importe quel nombre d'annotateurs et sur tout type d'annotation (ordinal ou non). Le calcul de l'alpha est plus compliqué que celui du κ de Cohen, je ne vais donc pas le préciser ici. Cependant, si vous voulez en savoir plus, je vous conseil l'article de Krippendorff de 2011 [2].

Krippendorff considère qu'un accord est bon seulement si $\alpha \geq 0.8$.

Chapitre 3

Présentation du travail réalisé

3.1 Reprise outil calcul des métriques

La première étape du stage était de reprendre un outil permettant le calcul des métriques κ, α et π, développé en C++ par Jeanne Villaneau.

Le programme prend en entrée un fichier .csv contenant le nombre d'annotateurs, le nombre de classes et le nombre d'observables, ainsi que le tableau des annotations. (chaque ligne i de ce tableau correspond à un annotateur et chaque colonne j un observable ; la case i,j correspond à l'annotation qu'à attribué l'annotateur i à l'observable j). Il peut également y avoir des annotations manquantes, qui sont représentées par un point.

Le programme commence donc par lire un de ces fichiers en entrée, en enregistrant toutes ces infos, puis crée un tableau de coïncidence qui est utile dans le calcul de l'alpha.

Ensuite, le programme calcul la métrique choisi, et nous affiche le résultat.

Après avoir pris connaissance de cet outil et d'avoir compris son fonctionnement, la prochaine étape consistait à convertir les données qu'ils ont réalisées précédemment (les corpus opinion, émotion et coréférence) en fichier qui puissent être lu par le programme.

Les données d'opinion et d'émotion étant dans un tableau Excel, j'ai donc créé un programme permettant de récupérer les données de ces tableaux et de créé un .csv de la forme lisible par le programme de calcul de métriques. Ces deux corpus font 25 annotateurs chacun, il a été choisi au début de partir sur des groupes de 9 annotateurs : j'ai donc divisé les corpus en trois (1-9, 9-17, 17-25).

Pour le corpus coréférence, les données était dans plusieurs fichiers composés de balise. J'ai donc réalisé un programme permettant d'extraire les informations qui nous intéressait dans ces fichiers (en utilisant certaines balises pour se repérer), puis de mettre ces données au format lisible par le programme de calcul de métriques.

3.2 Interprétation concrète des valeurs des métriques

3.2.1 Problématique

Lorsque ces métriques sont utilisé, on obtient une valeur allant de 0 à 1 représentant l'accord entre les annotateurs. Seulement, il peut être difficile de définir ce qu'est un bon accord : Carletta (1996) estime qu'un accord supérieur à 0.67 avec le κ de Cohen est un accord suffisant, tandis que d'autre comme Krippendorff estime qu'un accord de 0.67 avec le κ de Cohen est un accord moyen, et qu'un accord devient bon qu'à partir d'un κ de 0.8. L'interprétation de ces mesures reste donc un problème ouvert : l'objectif principal de ce stage sera d'essayé de répondre à ce problème.

Pour cela, il est question d'essayé de répondre à cette question :

avec N annotateurs, quel est le pourcentage de chance que la référence observé soit la référence idéale, en fonction du nombre de classe et de la valeur de la métrique ?

La référence observé étant les votes majoritaires obtenu sur chaque observable avec k annotateurs, et la référence idéale étant les votes majoritaires obtenus sur chaque observable avec le maximum d'annotateurs possible.

On sait que le nombre de catégories influe sur l'estimation obtenue : moins il y a de catégories, plus le κ est élevé. C'est pourquoi il est pris en compte au même titre que la valeur de la métrique dans la question.

Pour bien comprendre les enjeux de répondre à cette question et en quoi cela nous permet de définir l'interprétation des valeurs des métriques, on va prendre un exemple :

On pourra dire par exemple qu'avec un κ de 0.8 et 3 classes, avoir 5 annotateurs nous permet d'avoir les mêmes votes majoritaires à 5% près qu'avec 25 annotateurs, tandis qu'avec un κ de 0.6 on n'aurait les mêmes votes majoritaires qu'à 15% près. Cela permet donc d'avoir un résultat concret lié à la valeur de la métrique, qui nous permet de dire que dans tel cas, on considère que 5% d'erreur est tolérable mais que 15% ne l'est pas : on considérera donc que dans ce cas on obtient un bon accord à partir d'un κ de 0.8.

3.2.2 Méthode

Le travail à réaliser pour répondre à cette question consiste à obtenir en sortie d'un programme un tableau avec le pourcentage de modification par rapport à la référence en ordonnée et le nombre d'annotateurs en abscisse, et cela pour une certaine valeur de κ ou de α, et un certain nombre de classes.

Ce pourcentage de modification est calculé de la façon suivante :

-On calcul les votes majoritaires pour chaque observable de notre référence (dans notre cas, la référence sera soit avec 9 annotateurs, soit avec 25). On note le nombre d'annotateurs de notre référence n.

-Pour chaque nombre k$<$n d'annotateurs, on calcul les votes majoritaires pour chaque observable de chaque combinaison possible de $\binom{n}{k}$. Pour chacune de ces combinaisons, on compare les votes majoritaires obtenus par rapport à ceux de la référence, on comptant ceux qui sont différents : cela nous permet

d'avoir un pourcentage de votes majoritaires différents de la référence pour chaque combinaison.

-On fait ensuite la moyenne de ces pourcentages sur l'ensemble des combinaisons de $\binom{n}{k}$ pour obtenir le pourcentage de votes majoritaires différents de la référence avec k annotateurs.

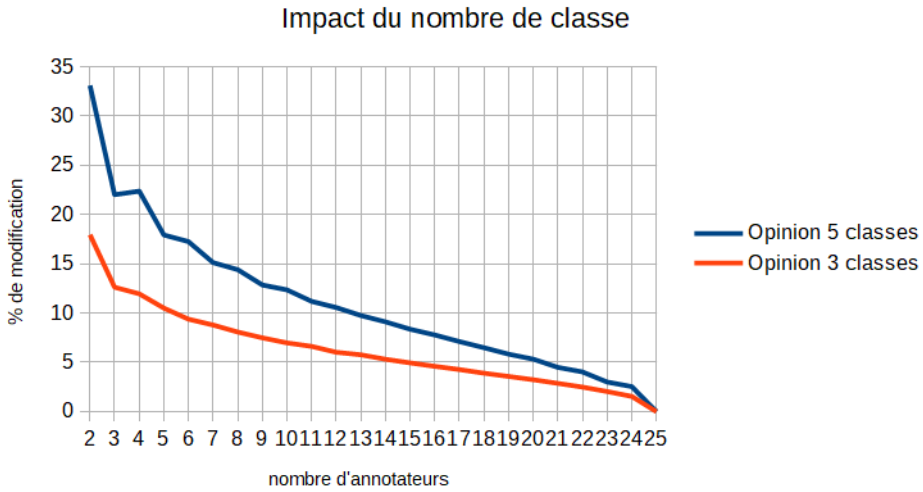
La principale difficulté était que le nombre de k-combinaisons pour tous les k correspond au nombre de sous-ensembles d'un ensemble de n éléments, c'est-à-dire 2^n. Dans notre cas, on commence avec k=2 (car on ne peut pas calculer d'accord avec un seul annotateur), ce qui nous fait :

$\sum_{k=2}^{n} \binom{n}{k} = 2^n - n - 1$

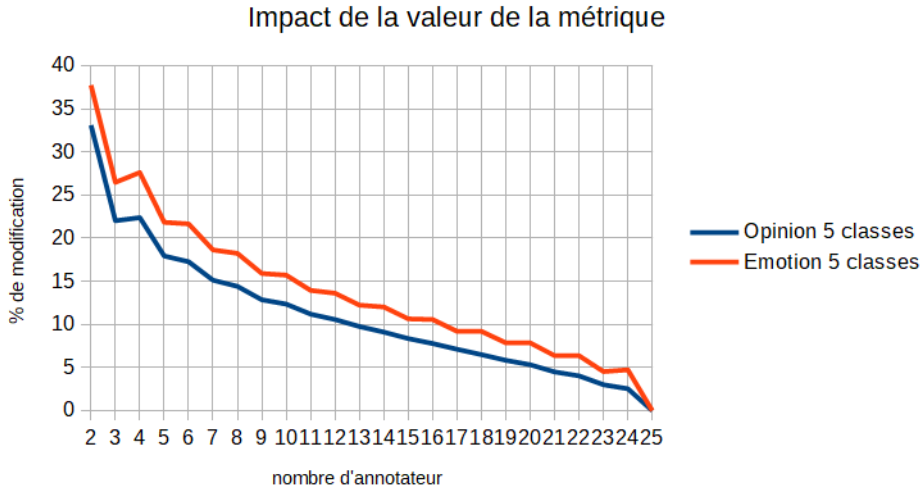
Ce qui reste du $O(2^n)$. Cette contrainte empêche donc d'essayé avec une référence très grande, et nécessite déjà d'être assez optimisé pour fonctionner avec une référence à 25 annotateurs.

3.2.3 Résultats

Ces résultats ont été effectué sur les corpus opinion et émotion, avec une référence à 25 annotateurs. Ces corpus étant à la base sur 5 classes (très négatif, négatif, neutre, positif, très positif), ils ont également été mis sur 3 classes afin de pouvoir regarder si le nombre de classe à bien un impact sur les résultats : pour cela, l'intensité des classes a été enlevé, ne restant que la polarité (négatif, neutre, positif).



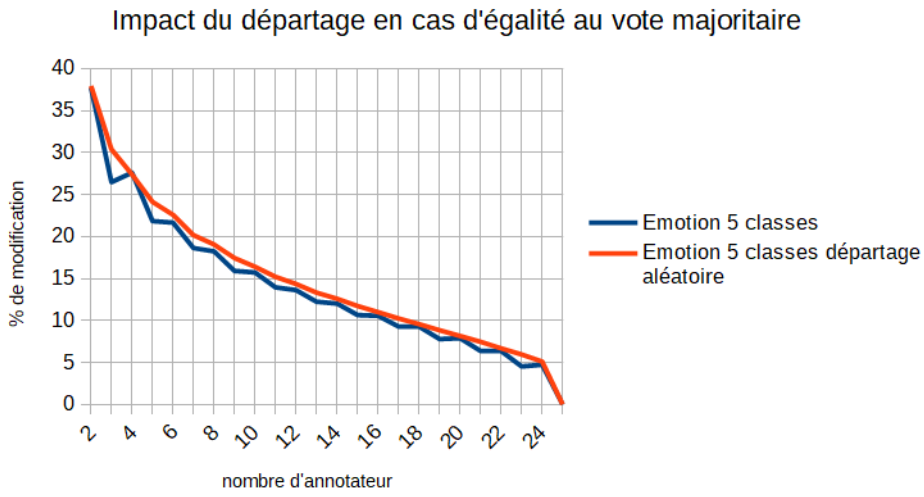
Ce premier graphique nous permet de mettre en évidence l'impact du nombre de classe sur le pourcentage de modification. Les résultats sont assez attendu : plus il y a de classe, plus il y a de chance que les annotateurs ne soient pas d'accord. Or étant donné qu'un vote majoritaire à négatif sur un observable alors que la référence à un vote majoritaire à positif sur cet observable est considéré comme étant le même nombre de modification que si le vote majoritaire était à neutre sur cet observable, il est normal que le pourcentage de modification augmente lorsqu'il y a plus de classes. Plus tard dans le stage, des tests seront fait en pondérant ces différences entre classes.



Ce deuxième graphique nous montre la différence entre le corpus opinion et émotion : opinion ayant un κ de 0.58 et émotion seulement de 0.3, on observe qu'avoir une plus grande valeur de kappa diminue le pourcentage de modification. Encore une fois, c'était le résultat attendu, l'objectif de ce stage étant d'interpréter la valeur des métriques par rapport à ce pourcentage de modification.

Ces deux résultats nous confirme donc que nos résultats finaux devrons effectivement être pour une certaine valeur de κ ou de α, et un certain nombre de classes.

On observe également d'étrange plateaux sur émotion 5 classes. après plusieurs tentatives d'explications de ces plateaux, j'ai fini par trouver la raison :



Sur le graphique ci-dessus, on peut voir qu'il y a des plateaux qui regroupe les groupes d'annotateurs par 2. En cas d'égalité au vote majoritaire, on départage en choisissant comme vote majoritaire la classe la plus proche de la moyenne

des classes de cet observable. L'idée derrière ce choix est de se dire que la classe la plus proche de la moyenne est plus légitime d'être le vote majoritaire si elle est plus proche de la moyenne (par exemple un cas où il y a 10 annotations de classe 0, 10 annotations de classe 1 et 5 annotations de classe 2, on choisi 1).

Même si l'idée paraît bonne, elle ajoute en fait un biais qui est la cause des plateaux. En effet, même en ajoutant un annotateur à un groupe, le vote majoritaire pouvait passer de strictement majoritaire à égalité avec une autre classe, mais cet égalité faisait que le vote majoritaire restait le même. Comme on peut le voir dans le graphique ci-dessus, en remplaçant ce départage en cas d'égalité au vote majoritaire par un choix aléatoire entre les classes à égalité, les plateaux disparaissent.

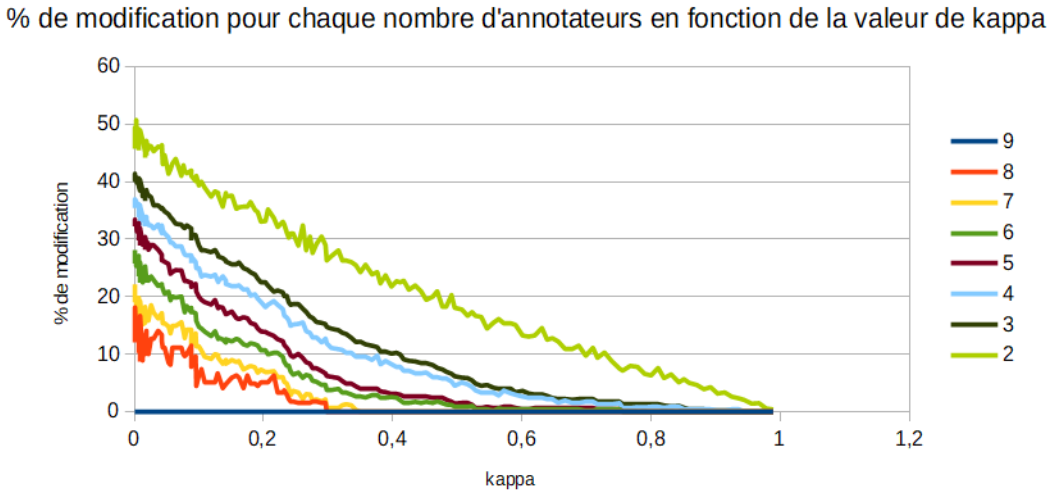
Ces résultats sur les données réelles que nous avons à disposition sont plutôt encourageant, mais nous n'avons des résultats que sur quelques valeurs de métrique. La prochaine étape est donc de créé un outil de génération aléatoire, afin d'avoir des valeurs pour toutes les valeurs de métriques.

3.3 Création outil génération aléatoire de données

Cette étape consiste à développer un code de modification aléatoire de données réelles qui permettra de calculer des mesures d'accord sur des données d'annotation dont on aura contrôlé la variation.

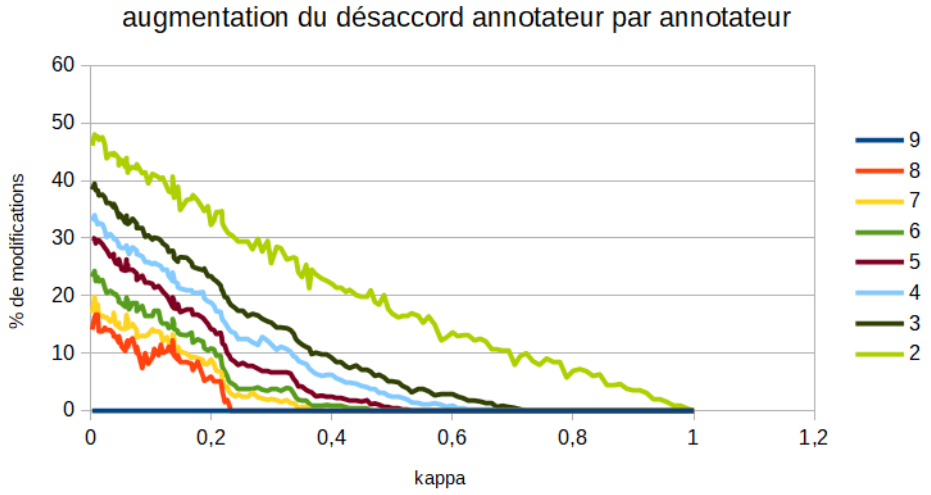
3.3.1 tentatives de générations aléatoire

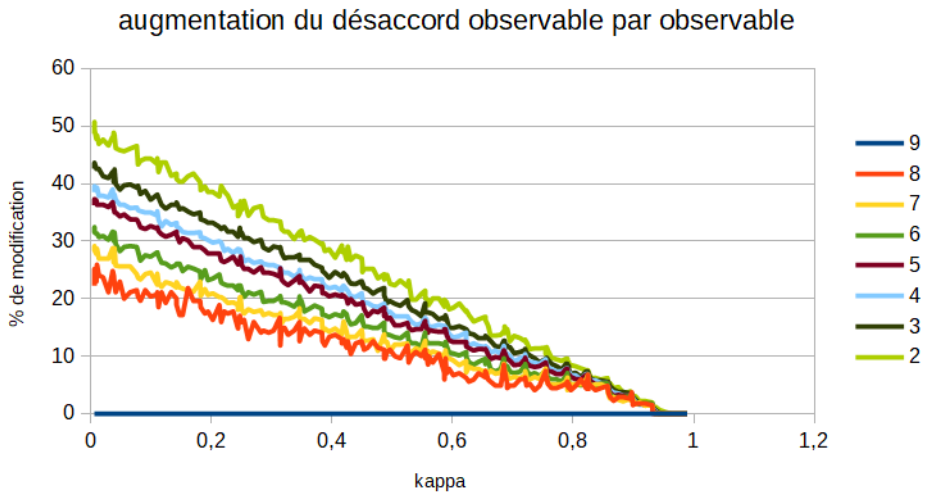
La première méthode que j'ai essayé pour générer aléatoirement des données était de partir de données où tout le monde est d'accord avec tout le monde, puis de choisir les annotations une à une dans un ordre aléatoire pour les modifier aléatoirement (en pouvant retomber sur la même classe). Dans toute cette partie, les tests ont été effectués avec une référence à 9 annotateurs. Voici les résultats de cette méthode :



On remarque que par rapport aux résultats que l'on obtient avec des vrais annotations, les pourcentages d'erreurs pour les groupes de 3 à 8 annotateurs sont de 0% jusqu'à des valeurs de kappa assez basses (par exemple les groupes de 8 annotateurs qui restent à 0% jusqu'à un kappa de 0.23, alors que pour tout nos résultats avec des kappa aux alentours de 0.6 ce n'est pas le cas).

On observe donc que cette méthode ne permet pas de simuler des données réelles. Néanmoins, avant de passer à une autre méthode, j'ai essayé de comprendre pourquoi on obtenait ces résultats :





ici, on a les même paramètres que dans l'exemple précédent, à la différence près que au lieu de choisir les annotations une à une dans un ordre aléatoire, je les choisi dans un certain ordre :

soit pour chaque annotateurs je modifie leurs annotations une à une, soit pour chaque observables je modifie les annotations qu'on leur a donné une à une.

Le premier cas correspond donc pour des valeurs de kappa élevé au cas où la majorité des annotateurs sont totalement d'accord entre eux et une minorité est totalement en désaccord avec tous les autre, et le second cas correspond à tous les annotateurs sont totalement d'accord sur la majorité des observables mais totalement en désaccord sur une minorité d'observable.

En observant les résultats, on observe que dans le premier cas les pourcentages restent à zéro encore plus longtemps, et dans le second cas à l'inverse les pourcentages sont supérieurs à zéro très rapidement.

J'en déduis que nos annotations sont plus proches du second cas que du premier cas, mais j'imagine que ce ne sera pas toujours le cas pour tous les corpus, puisque du coup ça dépend de la difficulté des observables et des personnes qu'on choisi pour annoter.

3.3.2 méthode choisi

La solution qui a été choisi pour généré des données consiste à partir de données réelles, et de créer de nouveaux annotateurs en fonction du comportement des vrais annotateurs. Pour obtenir des résultats pour toutes les valeurs de métriques, on pourra faire varier le nombre d'erreurs de nos nouveaux annotateurs. Pour cela, on va avoir besoin du nombre d'erreurs par observable (une erreur étant une annotation qui n'est pas de la classe majoritaire) qu'on appellera p1, et on appellera p2 le nombre d'erreurs des annotateurs.

Pour le nombre p1, on étudie ce qui se passe pour chaque observable : on calcule le vote majoritaire pour cet observable, ainsi que le nombre d'annotateurs qui n'ont pas choisi cette classe majoritaire pour cet observable.

Pour p2, on étudie ce qui se passe pour un annotateur : on calcule la classe majoritaire pour l'ensemble des observables, puis le nombre de fois où l'annotateur a_i, sur l'ensemble des observables, n'a pas choisi la classe majoritaire. La somme des erreurs de chaque annotateur diviser par le nombre d'annotateurs nous donne la moyenne du nombre d'erreurs par annotateur.

L'idée, c'est de pouvoir créer des annotateurs fictifs plus ou moins "bons", c'est-à-dire qui s'écartent plus ou moins de la référence par rapport à nos annotateurs réels, ce qui devrait nous permettre de pouvoir obtenir des valeurs d'accords plus ou moins élevées suivant la façon dont on déciderait de faire varier ce paramètre. Si on choisit pour un groupe d'annotateurs fictifs, un taux d'erreurs globalement $<$ à p2, on verra augmenter la mesure d'accord, alors que dans le cas contraire, on la verra baisser.

ensuite, on fait le protocole suivant : 1 - on choisit, en fonction du taux d'accord que l'on souhaite, un intervalle dans lequel on veut faire varier le taux d'erreurs par annotateur. Par exemple, si on a en moyenne p2=20% sur les observateurs réels et qu'on a 100 observables, on peut choisir [10 ;25] si on veut améliorer l'accord et [15 ;30] si on veut au contraire le dégrader.

2- Pour chaque annotateur fictif créé, on tire au sort le nombre d'"erreurs" (nbe) qu'il va commettre (par rapport aux votes majoritaires calculées avec les annotateurs réels) dans cet intervalle.

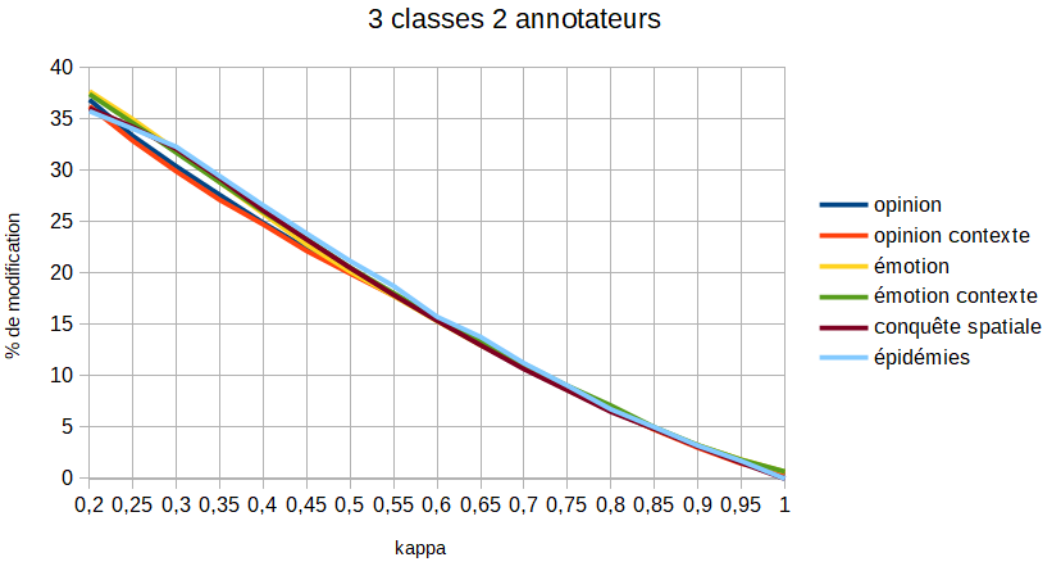
3 - on tire au sort nbe observations pour lesquelles l'annotateur fictif va choisir aléatoirement une classe différente du vote majoritaire, en tenant compte des erreurs constatées sur les observations (p1) : on a donc un tirage qui favorise les observations ayant eu le plus fort taux d'erreurs.

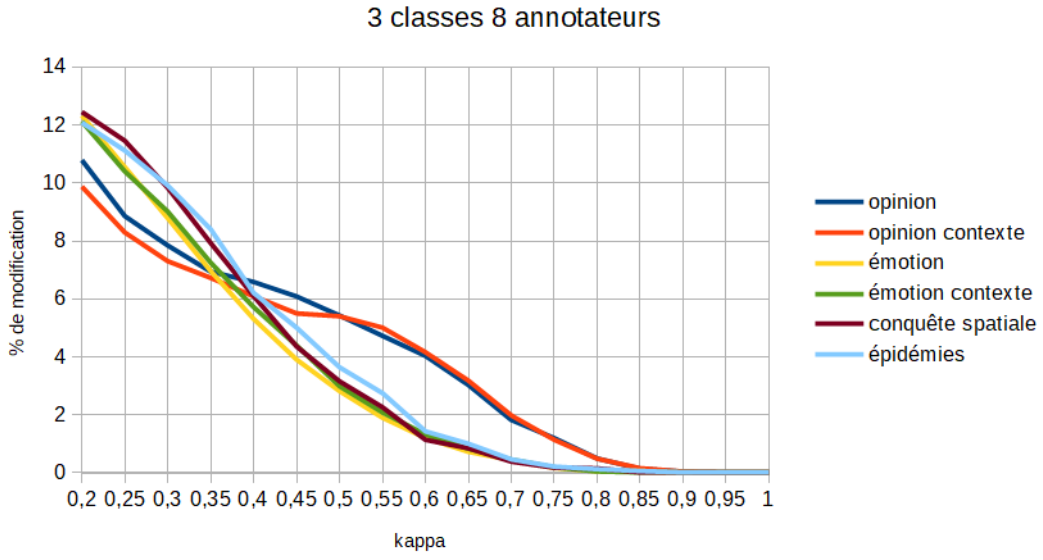
3.3.3 1er résultat de la génération aléatoire

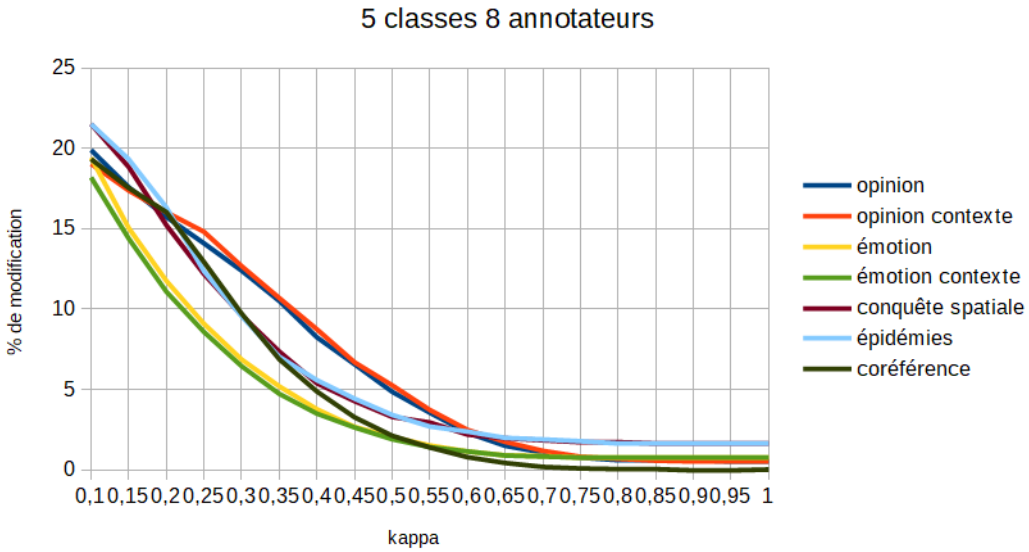
En appliquant cette méthode de génération aléatoire sur tous les corpus que nous disposions (opinion, émotion, similarité et coréférence), on obtient les résultats de l'annexe.

Ce sont les résultats de la génération aléatoire lancé 2000 fois sur chaque corpus, en ayant regroupé les résultats par tranches de 0.05 de valeur de Kappa, en calculant la moyenne des valeurs et l'écart type pour chaque tranches. Pour chacun des 2000 essais, ont créé 9 annotateurs avec un p2 (nombre d'erreurs par annotateur) différent pour chaque essai, puis on refait la même méthode que précédemment en calculant le pourcentage de modification pour toutes les combinaisons d'annotateurs de 2 à 8 parmi les 9 créés.

En regroupant les résultats de tous les corps sur les mêmes graphiques, on obtient les résultats suivant :







(les résultats de 3 à 7 annotateurs sont en annexe) On observe que sur les résultats pour 2 annotateurs, le pourcentage de modification est presque le même pour tous les corpus, et plus on a d'annotateurs, moins cela est vrai. Pour expliquer cela, j'ai émis l'hypothèse suivante : moins un groupe à d'annotateurs, moins il dépend de P1, donc moins il dépend des données de base (donc ses résultats sont plus stable). En effet, les pourcentages d'erreur des groupes avec beaucoup d'annotateurs ne peuvent changer que si plusieurs annotateurs font des erreurs sur les mêmes observables, et les chances que plusieurs annotateurs fassent des erreurs sur les mêmes observables dépend de P1 (il y aura plus de chance que ce soit le cas sur des observables avec beaucoup d'erreurs de base).

On remarque également dans ces résultats que le corpus opinion a un comportement différent des autres, en particulier pour 3 classes. De cela découle deux questions :

Pourquoi le comportement différent du corpus opinion ?

pourquoi des résultats très proches (hors opinion) pour 3 classes et pas pour 5 classes ?

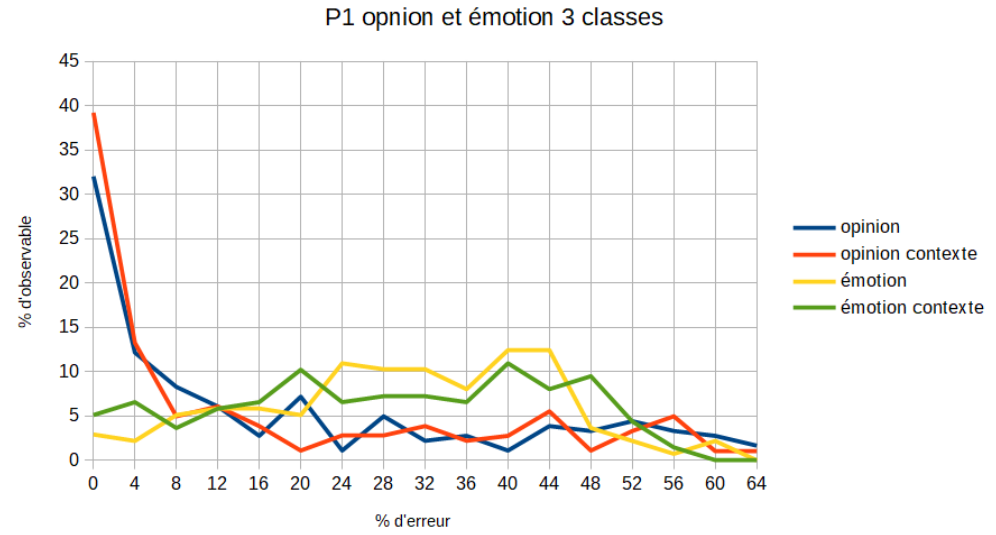
Quand on regarde les résultats un à un (annexe), on observe également que opinion 3 classes est le seul corpus pour qui ses résultats se regroupent par 2. C'est également une question à laquelle j'ai essayé de répondre.

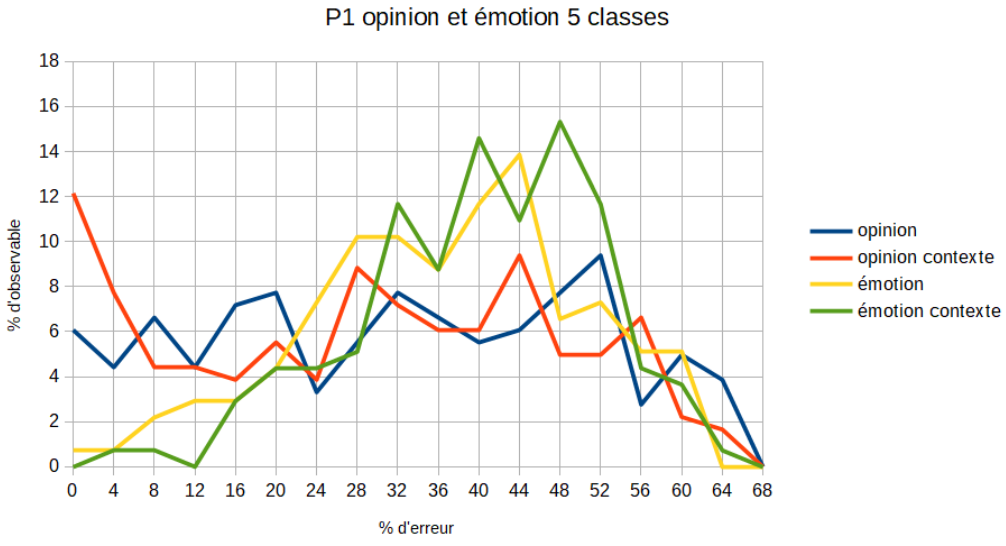
Pour essayé de répondre à ces questions, j'ai effectué plusieurs études sur des potentiels facteurs autre que le nombre de classe et la valeur de la métrique qui pourraient influencer les résultats.

Le premier test est sur la prévalence des classes. La prévalence est le nombre d'annotations de la même classe, la classe prévalente est la classe la plus représentée dans les annotations. Par exemple, si 60% des annotations sont de la classe 0, 30% de la classe 1 et 10% de la classe 2, la classe prévalente est la classe 0.

Pour cela j'ai échangé pour une certaine partie des observables les annotations de la classe prévalente avec les annotations de la classe la moins prévalente, afin de garder le même P1 (nombre d'erreur par observable) mais de changer la prévalence. On observe que les résultats sont un peu près identiques, donc la prévalence n'est pas un paramètre qui influence nos résultats.

Le deuxième test est sur la répartition des erreurs par observable (p1). Pour cela, j'ai observé cette répartition sur nos données réelles (ici opinion et émotion) :





On observe que opinion 3 classes est le seul à avoir un grand pourcentage d'observable n'ayant pas d'erreur. Après quelques tests en créant artificiellement des données ayant des répartitions d'erreurs par observables différentes (notamment en ayant un fort déséquilibre comme c'est le cas avec opinion 3 classes qui

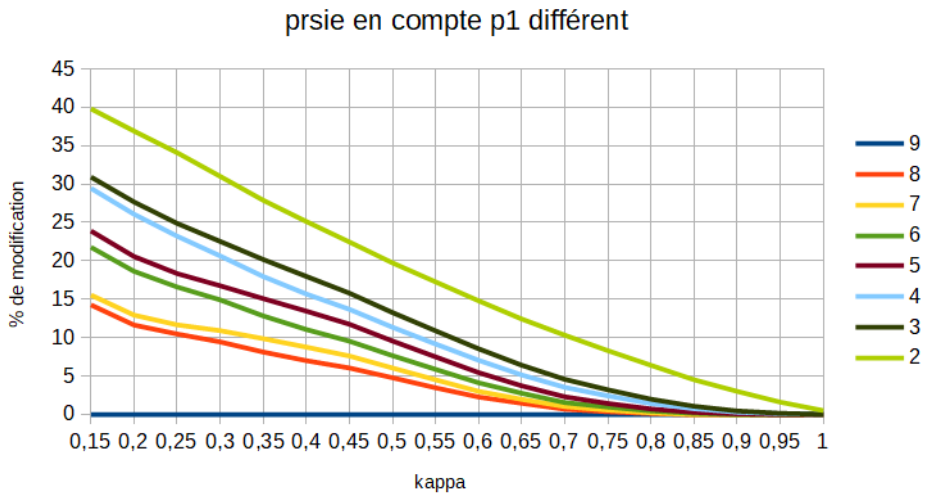
a 40% de ses observables où tout le monde est d'accord dessus), il semblerait que ce paramètre influence les résultats.

Le troisième test est sur l'homogénéité des annotateurs : jusqu'à maintenant, pour faire évoluer le nombre d'erreur par annotateur (P2), leurs pourcentages d'erreurs changent mais pas l'intervalle d'erreur, c'est-à-dire que les 9 annotateurs qui sont créés à chaque fois faisaient tous le même nombre d'erreurs. Ici, j'ai modifié ça en prenant au contraire l'intervalle le plus grand possible à chaque fois (par exemple s'ils doivent faire en moyenne 30% d'erreurs, ils feront entre 0 et 60% d'erreurs). On remarque qu'en procédant ainsi sur opinion 3 classes, les regroupements par deux disparaissent. Sinon, peu de changement sur les autres corpus.

J'ai également effectué quelques tests sur la méthode de génération aléatoire afin de voir si le problème pouvait venir de notre manière de générer les données.

Le premier test est que lorsqu'un annotateur généré fait une erreur sur un observable, la classe de l'erreur n'est plus totalement aléatoire mais est en fonction des erreurs faites sur cet observable sur les données réelles. On a des légères modifications des résultats mais rien de significatif.

Le second test est sur la prise en compte de p1 (le nombre d'erreur par observable) lorsqu'un annotateur créé fait une erreur. J'ai modifié la manière dont influençait p1 sur le choix de l'observable en cas d'erreur.



On observe que cela à un impact sur l'allure de la courbe, mais que les résultats sont néanmoins moins proche de ceux de nos données réelles.

Au final de tout ces tests, on conclue que p1 est également un paramètre qui a de l'importance, et que plusieurs autres facteurs tel que l'homogénéité des annotateurs ou la manière dont p1 influence les probabilités que les annotateurs fassent des erreurs sur les observables avec le plus d'erreurs influence aussi les résultats.

Par rapport à l'objectif, si en plus du nombre de classe et de la valeur de la métrique, on doit prendre en compte ces nouveaux paramètres, cela devient compliqué d'obtenir des tableaux généraux qui fonctionnerait pour tout. Cependant, les résultats semblent malgré tout exploitable, notamment pour les

groupes avec peu d'annotateurs, et montre bien ce que l'on voulait montrer à la base, c'est-à-dire des résultats qui permettent d'interprêter la valeur des métriques d'estimation de l'accord inter-annotateur.

3.4 divers

Graphique 7 : on revient sur opinion 3 classes avec cette fois-ci comme seul changement la métriques qui change : on utilise l'alpha pondéré (par une distance euclidienne). Très peu de changement.

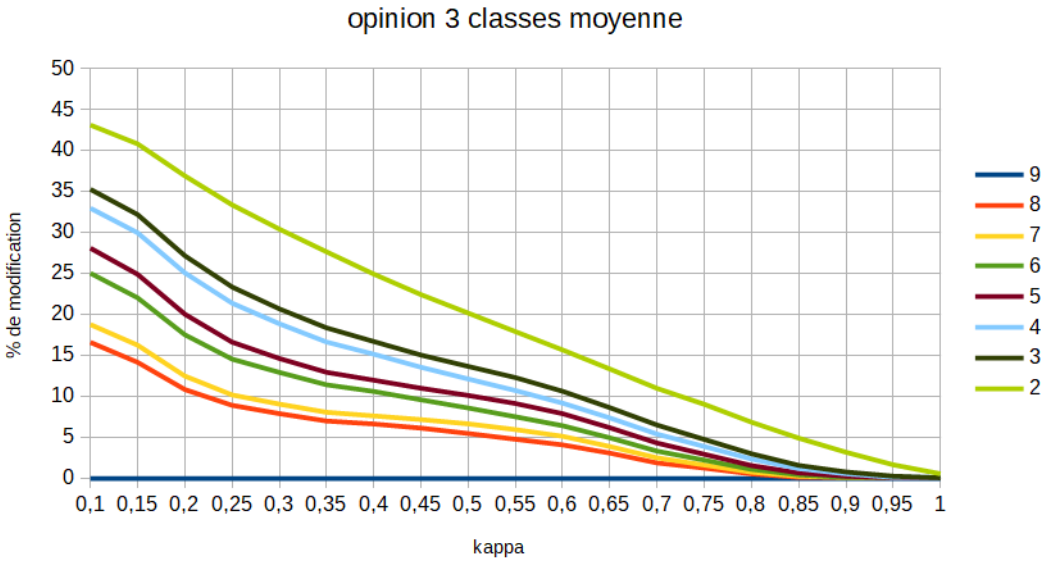
Graphique 8 : même chose mais cette fois-ci l'erreur aussi est pondéré : par exemple pour 3 classes répondre 0 alors que le vote majoritaire est 2 compte comme 1 erreur, et répondre 1 compte comme 0.5 erreur. Les valeurs ont chuté (ce qui est logique puisque ça peut valoir moins qu'une erreur). Ce qui est intéressant c'est que l'on peut du coup le comparer avec opinion 5 classes (graphique 9) où on obtient des valeurs du même ordre de grandeur. (mais ça n'a pas l'air d'avoir d'impact sur l'allure des courbes)

Chapitre 4

Bilan

Annexe A

Résultats expérimentaux



Bibliographie

[1] Anaïs Lefeuvre Jean-Yves Antoine, Jeanne Villaneau. Weighted krippendorff's alpha is a more reliable metrics for multi-coders ordinal annotations : experimental studies on emotion, opinion and coreference annotation. juin 2014.

[2] K Krippendorff. Computing krippendorff's alpha-reliability. 2011.