GeoBench : un outil d'alignement entre entités spatiales pour la construction d'un benchmark cartographique

Anthony MORANA et Thomas MOREL
Université Claude Bernard Lyon 1
anthony.morana@etu.univ-lyon1.fr
thomas.morel2@etu.univ-lyon1.fr

Février 2014

Résumé

Face à l'augmentation considérable d'applications mobiles et géolocalisées, de plus en plus de fournisseurs de services utilisent des systèmes d'information géographique. Chaque fournisseur propose des points d'intérêts (POI) sous forme d'entités spatiales. Cependant, un même POI peut ne pas être représenté de la même manière entre deux fournisseurs cartographiques (e.g., incohérence des données, informations incomplètes). Des travaux récents sur l'alignement entre entités spatiales permettent d'établir des correspondances entre fournisseurs, par exemple pour corriger des informations erronnées sur une entité. Cependant, il n'existe pas de benchmark pour tester ces approches d'alignement. Aussi nous proposons dans ce rapport un outil qui permet de construire un benchmark pour tester la pertinence des approches d'alignement. Pour faciliter la construction semi-automatique de ce benchmark, nous avons développé une approche d'alignement spatial, qui découvre les correspondances les plus probables. Nous avons validé expérimentalement notre approche avec trois fournisseurs sur un benchmark comportant quelques centaines d'entités spatiales.

Mots clés : alignement spatial, blocking spatial, GeoBench [1], benchmark, point d'interêt topographique, fournisseur de services géolocalisés

1 Introduction

Les services de géo localisation (LBS) sont aujourd'hui utilisés dans de nombreuses applications. Les fournisseurs cartographiques jouent un rôle essentiel dans la visualisation de points d'interêts (POI) (e.g., restaurants, hôtels, lieux touristiques). Un fournisseur a l'habitude de représenter un POI en utilisant des entités spatiales. La diversité et les multiples interconnexions entre les fournisseurs cartographiques entraine du bruit au niveau des données [DPS98]. Deux fournisseurs peuvent avoir des données incomplètes et/ou contradictoires pour le même POI. Ce bruit a un impact négatif quand des utilisateurs ont besoin de trouver des informations fiables et pertinentes.

1. Outil disponible à l'adresse suivante : http ://geobench.olympe.in/ (login : unimap – mdp : nautibus)

Afin de résoudre ce problème, des approches d'alignement entre entités spatiales qui se réfèrent au même POI ont été proposées [SKS+10]. Certaines approches considèrent seulement les informations spatiales pour détecter des correspondances alors que d'autres exploitent de multiples critères (e.g. nom, type, adresse) pour calculer une similarité entre deux entités spatiales [Olt07]. De telles approches requièrent une étape de validation, traditionnellement réalisée par des expériences sur les données du monde réel. À notre connaissance, il n'y a pas de benchmark pour évaluer l'alignement entre entités spatiales. Cette absence de benchmark ne facilite pas une comparaison equitable des différentes approches d'alignement. Cependant, construire un jeu de données expertisé est coûteux en temps et en effort humain. Nous estimons également utile l'expertise sur les attributs d'un ensemble de données, à la fois pour comprendre en quoi une approche d'alignement d'entités est efficace ou pas, et l'apprentissage.

Pour ces raisons, nous pensons qu'un benchmark pour l'alignement d'entités spatiales est nécessaire. Pour faciliter la construction d'un benchmark, il est nécessaire de proposer des correspondances à l'utilisateur, qu'il pourra ensuite valider. Après une étude des pré-requis (Section 2) et des travaux existants (Section 3), nous présentons deux algorithmes : un pour suggérer des correspondances entre des entités spatiales et un pour détecter les différences entre les entités correspondantes (Section 4). Par ailleurs nous décrivons notre outil GeoBench qui implémente les algorithmes mentionnés ci-dessus et qui aide les utilisateurs à créer un benchmark en validant les correspondances suggérées. Des expérimentations démontrent la qualité obtenue par nos algorithmes (Section 5).

2 Pré-requis

Un point d'intérêt (POI) est un lieu nommé du monde réel (e.g., la basilique de Fourvière). Dans notre contexte, un POI est représenté par un point, et nous ne traitons pas les POI représentés sous forme de polygone (e.g., campus de la Doua, parc de la Tête d'Or).

Une entité spatiale possède un ensemble d'attributs. On distingue les attributs primaires (identifiant relatif au fournisseur, coordonnées, type et nom) qui sont obligatoires des attributs secondaires (e.g. adresse, numéro de téléphone et site web). Par exemple, l'entité «basilique de Fourvière» chez Geonames a pour identifiant 8015555, pour nom «Notre-Dame de Fourvière», pour type «spot, building, farm» et «church», et pour coordonnées «45.76228, 4.82196»

La typologie des différences entre les données de deux entités correspondantes permet de qualifier un jeu de données et de faire de l'apprentissage. Deux entités issues d'une correspondance peuvent avoir des différences aux niveaux de la sémantique, des données spatiales, des schémas (attributs). En ce qui concerne les erreurs de sémantique, deux entités ont des données différentes au niveau de leurs attributs primaires et secondaires. Au niveau des erreurs spatiales, on fait la distinction entre différence de localisation (les deux entités possèdent des coordonnées spatiales différentes), position équivalente (les deux entités ont des coordonnées spatiales différentes mais réfèrent au même POI) et superposition (les deux entités ont les mêmes coordonnées mais correspondent à deux POI différents). En ce qui concerne la différence de schémas, on distingue la différence de structure et l'hétérogénéité des attributs. Dans le premier cas, un attribut de l'une des entités correspond à deux ou plusieurs attributs d'une autre entité. Dans le deuxième cas, deux attributs de deux entités différentes correspondent mais ont des labels différents. Enfin on considère également en erreur le fait qu'un POI n'est pas présent dans un ou plusieurs jeux de données.

3 Travaux similaires

Récemment, des travaux de recherche se sont intéressés à l'alignement d'entités spatiales. Nous discutons ensuite des benchmarks pour évaluer ces approches d'alignement.

3.1 Approches d'alignement spatial

Nous avons étudié l'article « Entity Resolution in Geospatial Data Integration » de Vivek Sehgal qui traite des différentes méthodes d'unification de points entre plusieurs sources de données [SGV06]. Pour effectuer les comparaisons, deux approches sont présentées. L'une consiste à comparer les données spatiales (latitude et longitude), tandis que l'autre consiste à comparer les données non spatiales (nom du lieu, type du lieu, etc). Dans l'article, la similitude du nom d'un lieu peut être vérifiée par trois méthodes : distance de Levenstein, distance de Jaccard et distance de JaroWinkler. La première calcule le nombre d'opérations (addition, suppression et changement de caractères) qu'il faut pour passer d'une chaîne de caractères à une autre. La deuxième calcule le rapport du nombre de lettres communes uniques sur le nombre total de caractères uniques d'une chaîne de caractères. La dernière calcule la similarité entre deux chaînes de caractères en fonction du nombre de caractères correspondants, de la longueur de la chaîne et du nombre de transposition.

Pour la comparaison de coordonnées, il est préconisé de prendre l'inverse de la distance entre deux points. En ce qui concerne la comparaison de type, chaque fournisseur possède sa propre cardinalité de type. De plus, il peut exister des différences entre les niveaux de précision des types de chaque fournisseur (e.g., type « rivière » chez le fournisseur A et type « cours d'eau » chez le fournisseur B). De ce fait, il ne faut pas comparer les chaînes de caractères comme précédemment, mais comparer leur sémantique.

Grâce aux différentes expérimentations, l'article préconise de combiner les différentes méthodes afin d'obtenir de meilleurs résultats. Au final, on peut voir que la comparaison par coordonnées, nom et type est la meilleure. Cependant quelques points de l'article sont critiquables. Tout d'abord la comparaison entre entités utilise le nom, les coordonnées et le type alors que d'autres attributs comme les numéros de téléphone ou l'adresse peuvent être comparés et ainsi améliorer la pertinence des résultats. Par ailleurs on peut remarquer que le jeu de données est peu pertinent (par exemple 80% des entités correspondantes ont le même nom) et non disponible.

Dans un article de Beeri, une autre approche est présentée [SKS+10]. Il s'agit d'une approche séquentielle qui utilise un algorithme de jointure sur un jeu de données. L'inconvénient d'une telle méthode, c'est que lorsqu'une erreur survient, elle influence le reste des résultats. Ainsi les erreurs peuvent s'accumuler au fil des jointures et fournir à la fin des résultats médiocres.

3.2 Benchmark pour l'alignement spatial

De nombreux travaux sur l'alignement d'entité comme celui de Sehgal ont été proposé. Un des points commun à l'ensemble de ces recherches : aucun des benchmarks présenté n'est disponible (parfois pour des raisons de confidentialité). On peut trouver de temps en temps des jeux de données [2] ou des vidéos [3] de présentation de benchmark sur internet mais rien de véritablement exploitable. Par exemple, le jeu de données sur les restaurants est limitée pour le défi de l'alignement, puisqu'une simple mesure d'égalité sur le numéro de téléphone permet

2. Jeu de données sur des restaurants présentant 112 corespondances correctes : http ://www.cs.utexas.edu/users/ml/riddle/data.html

3. http ://linqs.cs.umd.edu/projects/geoddupe/

de détécter toutes les correspondances correctes.

4 Notre approche

De notre côté, nous proposons un outil pour construire un benchmark qui s'appuie sur un algorithme d'alignement d'entités spatiales pour suggérer les correspondances les plus pertinentes à l'utilisateur. Pour trouver ces correspondances entre entités, nous procédons dans un premier temps à une phase de blocking puis à une phase d'alignement. Avec les résultats de ces algorithmes nous procédons à une détection des erreurs afin d'attribuer une valeur de similarité à chaque résultat. Pour finir, seuls les résultats les plus pertinents sont proposés à l'expert grâce à un algorithme de tri.

4.1 Algorithme du blocking

L'interêt du blocking est de retourner des correspondances entre une entité d'un fournisseur source et des entités d'autres fournisseurs. Pour cela, nous utilisons les API de chaque fournisseurs qui nous permette de comparer les entités (la pertinence des resultats est effectuer par chaque fournisseur) selon les attributs suivants : le nom, le type et les coordonnées.

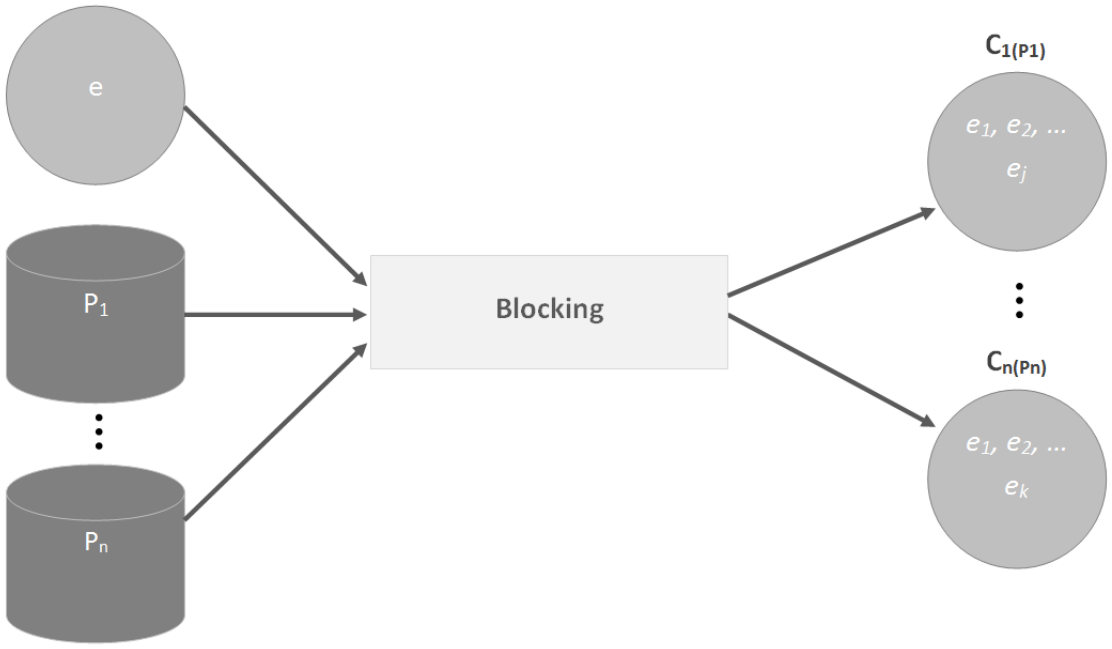

FIGURE 1 – Principe du blocking

La figure ci-dessus montre le principe de l'algorithme de blocking [4]. Il prend en entrée une entité source (symbolisé par le cercle gris clair) et un ensemble de fournisseurs (symbolisé par les cylindres gris foncé). En sortie, on obtient pour chaque fournisseur un ensemble d'entité en lien avec l'entité passé en entrée. La taille de ces ensembles peuvent être différents.

Chaque fournisseur met à disposition des URL afin de récupérer des données relatives aux entités. Pour récupérer les informations liées à la recherche effectuée, il est possible de

4. L'algorithme 1 (voir Annexe 1) présente l'algorithme de blocking

passer des paramètres à ces URL pour filtrer les résultats. Ainsi pour trouver des entités correspondantes avec l'entité source servant de base à l'alignement, nous avons utilisé comme paramètre le nom, le type et les coordonnées géographiques. Afin d'obtenir plus de pertinence dans les résultats, nous avons effectué deux requêtes chez chaque fournisseur. La première retourne les résultats ayant la même sous-chaine au niveau du nom et les mêmes coordonnées (modulo un rayon) que l'entité source (ligne 3). La seconde renvoie les résultats avec le même type (dans une hiérarchie de types fixés et fournies) et les mêmes coordonnées géographiques (toujours selon un rayon) que l'entité source (ligne 4). Par ailleurs pour filtrer les résultats, nous avons appliqué un périmètre de recherche pour la seconde requête. Ce rayon de recherche s'adapte en fonction du type de l'entité cible. Par exemple nous avons choisi un rayon de 200m pour les restaurants. Plus les lieux associés à un type peuvent être étendus et plus le radius pour ce type est grand. Une fois que nous avons obtenu un résultat pour les deux requêtes, nous prenons l'union de ces résultats (ligne 5). Ainsi pour chaque fournisseurs nous avons un ensemble d'entités correspondantes avec l'entité source.

4.2 Algorithme d'alignement

L'algorithme d'alignement permet de découvrir des correspondances entre une entité cible d'un fournisseur source et une ou plusieurs entités chez d'autres fournisseurs cartographiques (à condition qu'il y a un résultat chez le fournisseur cible). Contrairement à l'algorithme de blocking (qui compare simplement et rapidement 3 attributs), l'algorithme d'alignement utilise sur tous les attributs des mesures de similarité plus sophistiquées mais aussi plus coûteuses.

Ainsi nous combinons une mesure de similarité appliquée à tous les attributs pour maximiser le score de pertinence entre toutes les paires d'entités. Pour chaque entité résultant du blocking, l'algorithme d'alignement calcule son score de pertinence entre [0,1] avec l'entité source. Un score égale à 0 signifie qu'il n'y a aucun lien entre les deux entités, tandis qu'un score de 1 signifie que les deux entités sont parfaitement identiques. Pour génerer ce score de pertinence, nous calculons une valeur de similarité entre les attributs equivalents des deux entités considérées.

La figure 2 montre le principe de l'algorithme d'alignement [5]. Il prend en entrée une entité source et l'ensemble des entités retourné par le blocking. En sortie, on obtient pour tous les fournisseurs un ensemble de tuples contenant la valeur de similarité de chaque paires.

Pour la comparaison des coordonnées géographiques, nous calculons la distance euclidienne entre les deux entités (ligne 4 à 8) . Si la distance est supérieure au rayon de la recherche, son score est ramené à zéro, sinon il diminue proportionnellement à la distance (ligne 9 à 14). Pour la comparaison du nom, nous calculons la distance de Levenshtein entre les deux chaines de caractères (ligne 15 à 17). Nous avons choisi d'utiliser une seule mesure sur ces attributs pour des raison de performances et pour éviter des poids à combiner. De plus la distance de Levenshtein nous parait la plus efficace. En effet d'après l'article de Seghal, différentes méthodes de comparaison de chaines ont été testées et les meilleurs résultats ont été trouvés en utilisant la distance de Levenshtein [SGV06]. En ce qui concerne la comparaison du numéro de téléphone et du site web, nous procédons de la même manière (ligne 24 à 29). Pour la comparaison de l'adresse, nous réalisons un petit travail préparatoire afin d'uniformiser le formats hétérogène des adresse. En effet, pour que les résultats soient pertinents, nous formatons toutes les adresses de la manière suivante : numéro de rue, rue, ville, pays. Ensuite nous pouvons appliquer la distance de Levenshtein (ligne 19 à 23). En ce qui concerne la comparaison du type, nous ne pouvions pas réaliser de rapprochement avec la distance de

5. L'algorithme 2 (voir Annexe 3) présente l'algorithme d'alignement

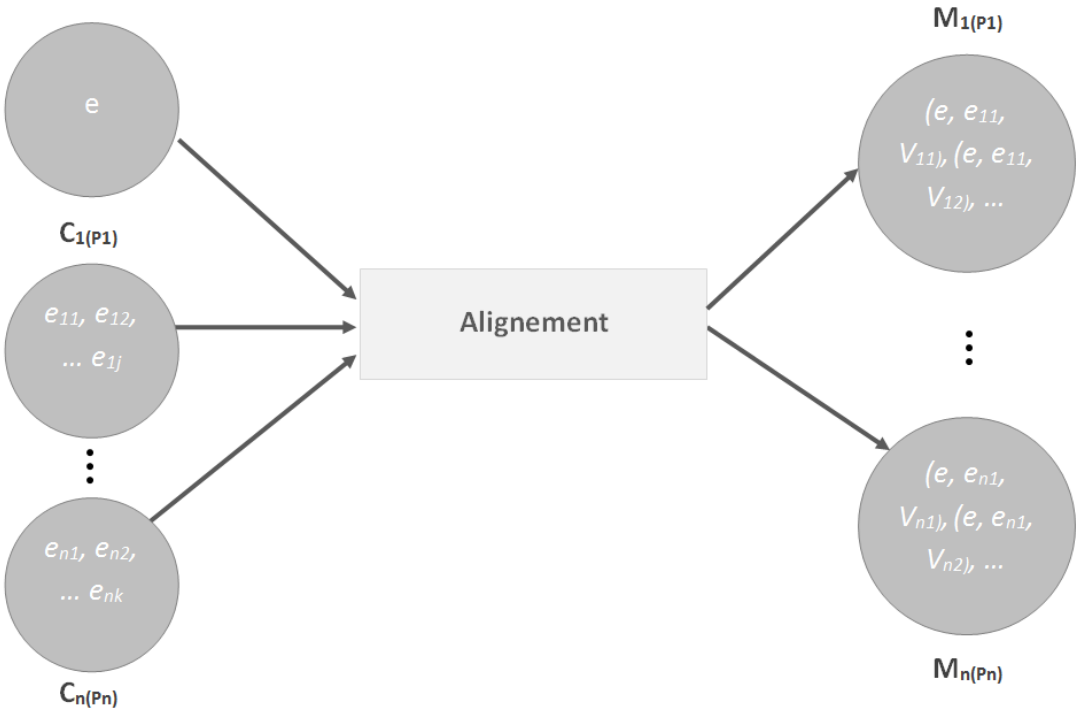

FIGURE 2 – Principe de l'alignement

Levenshtein. En effet, il ne faut pas comparer la syntaxe des chaines, mais leur signification. Pour cela nous utilisons une taxonomie « Unimap » de correspondances les types des différents fournisseurs. Celle-ci est fournie par nos encadrants et a été réalisé manuellement dans la cadre du projet Unimap. Si le type de l'entité source correspond au type de l'entité cible alors on attribue le score maximum, sinon on donne la note de zéro (ligne 18).

Une fois que l'on a calculé une valeur de similarité pour chaque attribut, on fait la moyenne pondérée de ces notes (figure 3). Pour des raisons d'efficacité, nous avons attribué un poids plus important aux valeurs qui portent sur les attributs primaires (coordonnées, nom et type). Ces trois renseignements sont par ailleurs toujours mentionnés par les fournisseurs. En revanche l'adresse, le numéro de téléphone et le site web peuvent être absents des données renvoyées par un fournisseur (par exemple Geonames ne fournit pas ces informations). Ainsi nous attribuons un poids de 2/3 aux attributs primaires et un poids de 1/3 aux autres attributs dans le calcul du score de pertinence entre deux entités (ligne 30).

4.3 Algorithme de tri

Une fois que chaque entité possède un score, nous trions celles-ci par ordre croissant de pertinence. Par ailleurs nous aurions pu affiner le triage en ne récupérant que les résultats de même type. Cependant par manque de temps, cette amélioration nous est apparue superflue.

4.4 Algorithme de différences

Cet algorithme permet de renseigner la typologie des erreurs. Le but de cette typologie est de recenser les différentes erreurs qui peuvent survenir au niveau des correspondances (faux positifs et faux négatifs) durant un processus de découverte de correspondances. Un faux positif est une correspondance découverte par une approche d'alignement qui n'aurait

Attributs	Entité source	Entité cible	Valeur de similarité
Coordonnées	45,762293 ; 4.822626000000014	45.7623 ; 4,82155	0,871
Nom	La Basilique Notre Dame de Fourvière	Basilique de Fourvière	0,61
Type	church, placeofworship ,establishment	Lieu de culte	1
Adresse	8 Esplanade de Fourvière, Lyon, France	8 esplanade de fourvière Lyon France	0,76
Téléphone			1
Site Web			1

Le score de pertinence entre ces deux entité est de 0,85

FIGURE 3 – Calcul de similarité entre deux entités

pas dû être découverte (car elle est fausse). Un faux négatif est une correspondance correcte (attendue) mais qui n'a pas été découverte par l'outil.

Pour le moment nous avons seulement géré les différences sémantiques et les différences d'emplacement. Pour ces différences, nous avons considérer un seuil pour chaque attribut. Dans notre cas nous avons pris un seuil de 0,5 car nous avons remarqué que celui ci était le plus adapté.

5 Validation expérimentale

Pour obtenir un jeu de données à valider expérimentalement, nous avons conçu un outil permettant d'établir des correspondances semi automatiquement. Cet outil nommé GeoBench propose une interface web permettant de rechercher des POI à aligner. Une fois celui-ci choisi, l'utilisateur est guidé pour valider / invalider les correspondances, ainsi que les différences entre attributs (e.g. cordonnées, nom, type, adresse, telephone, site). L'interface a été réalisée en HTML5 / CSS3, et utilise en majorité le langage JavaScript. Nous avons utilisé les API JQuery et Bootstrap pour définir le visuel du site et faciliter l'implémentation de nos algorithmes. Nous avons travaillé sur un code le plus générique possible pour pouvoir à l'avenir ajouter plusieurs fournisseurs, modifier le fournisseur source ou encore ajouter des correspondances entre types d'entités (e.g. restaurant chez Google équivaut à eat-drink chez Here et REST chez Geonames). GeoBench a donc permis de générer un jeu de données cartographique [6], expertisé par les membres du projet Unimap.

5.1 Protocole

Le jeu de données mis à disposition possède trois parties importantes. Premièrement, il met en évidence la liste des entités utilisées pour établir les correspondances. Il faut ainsi faire la distinction entre les entités sources (qui possèdent des informations considérées comme correctes), et les entités cibles. Ensuite, le jeu de données contient la liste des différences sur les attributs. Enfin, le benchmark possède la liste des correspondances potentielles entre une

6. Jeu de données disponible à l'adresse suivante : http ://geobench.olympe.in/GeoBench_benchmark.sql (login : unimap – mdp : nautibus)

entité source et une entité cible donnée. Chacune d'entre elle définit sa validité, ainsi que les différences qui ont été détectés automatiquement par l'outil, et les différences corrigées par l'utilisateur. Le tableau ci-dessous présente quelques statistiques sur le jeu de données disponible en ligne.

Fournisseur	**Google Maps (source)**	**Here maps**	**Geonames**
Nombre entités	100	310	157
Nombre entités en France	≈ 80	≈ 260	≈ 100
Nombre entités en Europe	≈ 85	≈ 275	≈ 110
Nombre de correspondances correctes	-	68	31

FIGURE 4 – Statistiques générales du jeu de données

Pour évaluer le jeu de données, nous utilisons trois mesures : la mesure de Précision, de Rappel et de F-Mesure (Figure 4). La précision est définie comme étant le rapport entre les correspondances validées correctes (TP) sur l'ensemble des correspondances découvertes (TP + FP). Le rappel est quant à lui définit par le rapport entre les correspondances validées (TP) sur le nombre d'entités du fournisseur source (TP + FN). Enfin, la mesure de F-Mesure F est obtenue à l'aide de la formule suivante :

$$F = 2.\frac{précision.rappel}{précision + rappel}$$

		Expert	
		Correcte	**Incorrecte**
Outil	**Correcte**	True Positive	False Positive
	Incorrecte	False Negative	True Negative

FIGURE 5 – Classification des résultats

5.2 Evaluation de la qualité de l'algorithme d'alignement

La première expérience a pour objectif de vérifier la pertinence des algorithmes de blocking et d'alignement à l'aide du jeu de données précédent. Nous voulons savoir si l'algorithme de matching retourne l'entité correcte en premier, c'est-à-dire avec le score de pertinence le plus élevé. Pour cela, les trois mesures décrites précédemment ont été utilisées. Tout d'abord, le calcul des statistiques est effectué sur la première correspondance établie pour chacun des fournisseurs (Top-1). Puis il est effectué sur les deux premières correspondances (Top-2) et ainsi de suite jusqu'à Top-5.

Sur la figure 6, on constate que lorsque l'utilisateur vérifie uniquement la première correspondance avec chacun des fournisseurs, celle-ci est précise à près de 50%. Cependant, on peut remarquer que seulement 20% des résultats proposés par les différentes cibles comporte l'entité correspondante avec la source. Cette fois-ci en arrivant au top-6, on remarque que 85% des résultats comportent l'entité correspondante. Cependant, la précision vient à diminuer car des entités incorrectes apparaissent dans les différentes correspondances à traiter par l'expert. Le calcul de F-Mesure étant stable autour de 50% nous indique cependant que la précision vient à être compensée par le rappel et inversement lorsque le nombre de correspondances à valider augmente.

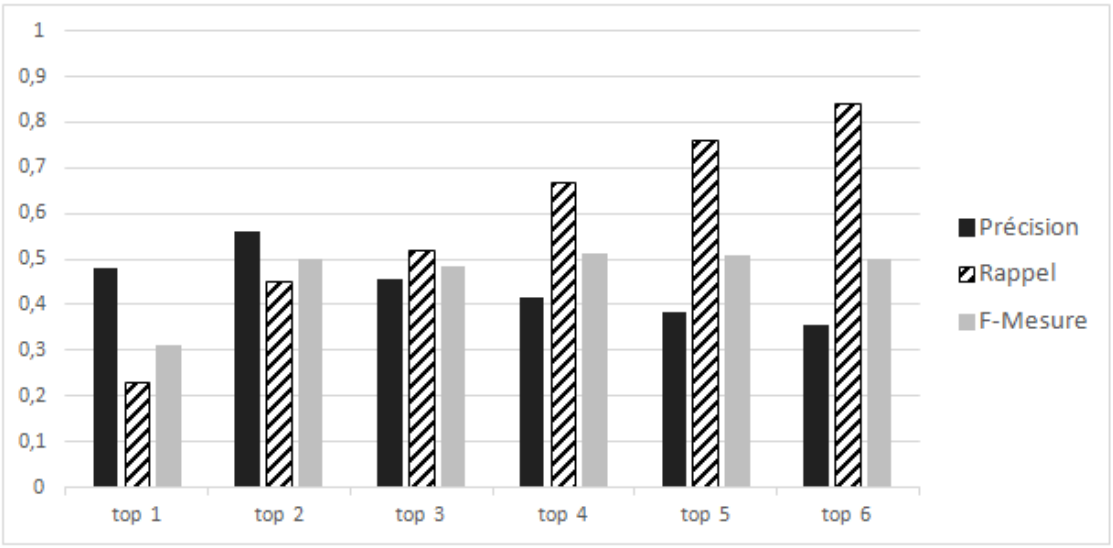

FIGURE 6 – Evaluation du jeu de données à l'aide des mesures de précision, de rappel et de f-mesure

5.3 Evaluation de la qualité de l'algorithme de détéction d'erreurs

Cette expérience vise à tester l'efficacité de l'algorithme des différences sur les attributs des entités source et cible. Ces résultats sont obtenus en comptant le nombre de différences obtenu sur chaque attribut automatiquement à l'aide de l'outil, puis ceux qui ont été corrigés par l'expert.

Erreur	Coordonnées	Nom	Adresse
Différences découvertes par l'outil	40	29	54
Différences corrigées par l'expert	39	28	54
Ratio	97,5%	96,55%	100%

FIGURE 7 – Ratio entre le nombre de différences trouvées par l'outil et le nombre de corrections effectuées par l'expert

La figure 7 présente le ratio entre le nombre de différences trouvées par l'outil et le nombre de corrections effectuées par l'expert. Ainsi l'outil a trouvé 40 différences au niveau des coordonnées. L'expert a corrigé une différence ce qui induit que notre outil est pertinent dans 97,5% des cas pour ce jeu de données. En ce qui concerne le nombre de différences au niveau du téléphone et du site web, nous n'avons pas présenté leurs résultats car ils n'étaient pas intéressants du fait que peu d'entités renseignent ces attributs.

6 Conclusion

Le but de ce projet était de fournir un outil d'aide à la construction d'un jeu de données expertisées. GeoBench permet de proposer des entités d'un fournisseur source (dans notre cas Google maps) et de trouver des correspondances avec d'autres fournisseurs. Ensuite l'expert peut valider ou invalider une correspondance. L'outil GeoBench repose sur un algorithme d'alignement et un algorithme de différences, qui ont été évalués sur un jeu de données réelles.

Pour arriver à ce résultat, nous avons rencontré quelques difficultés notamment avec Geonames. En effet, au début nous avions choisi ce fournisseur comme source de nos alignements. Cependant celui-ci ne dispose pas d'adresse, de téléphone et de site web pour ces entités ce qui a entrainé une diminution de la pertinence des résultats. De ce fait, nous avons dû changer notre fournisseur source pour Google maps. En ce qui concerne ce dernier, nous avons eu du mal à récupérer les données de ce fournisseur (Sans doute pour des raisons de sécurité). Par ailleurs la structure de chaque fournisseur cartographique est très hétérogène, ce qui nous a obligés de nous adapter. Pour cela nous avons du rendre notre outil générique afin de pouvoir récupérer les informations de n'importe quelle fournisseurs. De ce fait, notre application est plus modulable. Dans le cas d'une augmentation future du nombre de fournisseurs, il sera plus simple de faire évoluer l'application.

Avec plus de temps, nous aurions pu améliorer l'outil GeoBench. Par exemple lors de la recherche d'entité avec le fournisseur source nous pourrions faciliter la comparaison des résultats. De plus lors de la phase d'alignement, il n'est pas possible de revenir en arrière, ce qui peut fausser les résultats si l'expert commet une erreur. Par ailleurs si l'utilisateur n'est pas sûr d'une correspondance, il n'a pas la possibilité de consulter les autres, puis de revenir à celle qui n'a pas été validée. Enfin nous avons également une piste d'amélioration sur la comparaison des types. En effet, nous réalisons une comparaison trop stricte. On pourrait utiliser des mesures de similarité à l'intérieur de la taxonomie pour repérer des concepts plus ou moins similaire.

Ce TER nous a permis de nous familiariser avec une approche de recherche. Cela fut différent des autres projets où l'on ne prend pas le temps d'analyser le contexte.

Références

[DPS98] Thomas Devogele, Christine Parent, and Stefano Spaccapietra. On spatial database integration. *International Journal of Geographical Information Science*, 12(4) :335–352, 1998.

[Olt07] AM Olteanu. A multi-criteria fusion approach for geographical data matching. *International Symposion in Spatial Data Quality (ISSDQ)*, 2007.

[SGV06] Vivek Sehgal, Lise Getoor, and Peter Viechnicki. Entity resolution in geospatial data integration. In Rolf A. de By and Silvia Nittel, editors, *GIS*, pages 83–90. ACM, 2006.

[SKS+10] Eliyahu Safra, Yaron Kanza, Yehoshua Sagiv, Catriel Beeri, and Yerach Doytsher. Location-based algorithms for finding sets of corresponding objects over several geo-spatial data sets. *International Journal of Geographical Information Science*, 24(1) :69–106, 2010.

A Annexe 1

Cette annexe inclut notre algorithme de blocking, qui est détaillé en Section (4.1)

Algorithm 1 Blocking algorithm

Require: Une entite souce e
Require: Un ensemble de fournisseurs cible $\mathcal{P}$
Output : Des ensembles d'entités $\mathcal{C}_p$

```
1: function blocking()
2: for all p ∈ P do
3:     Γ_p ← get_entities(coords_e, type_e)
4:     Π_p ← get_entities(coords_e, label_e)
5:     C_p ← Γ_p ∪ Π_p
6: end for
7: end function
```

B Annexe 2

Emma se connecte à l'outil de benchmark en tapant l'URL dans son navigateur. Sur le premier écran, elle saisit les paramètres suivants : K=5 (chaque fournisseur proposera au maximum 5 correspondances potentielles) et elle recherche via Google Maps le POI « basilique de Fourvière ». Plusieurs entités Google Maps sont proposées pour sa recherche. Elle sélectionne celle identifiée par « c128fe99cb266174f8a50b94c99cbc13a7f58375 », qui correspond au POI recherché.

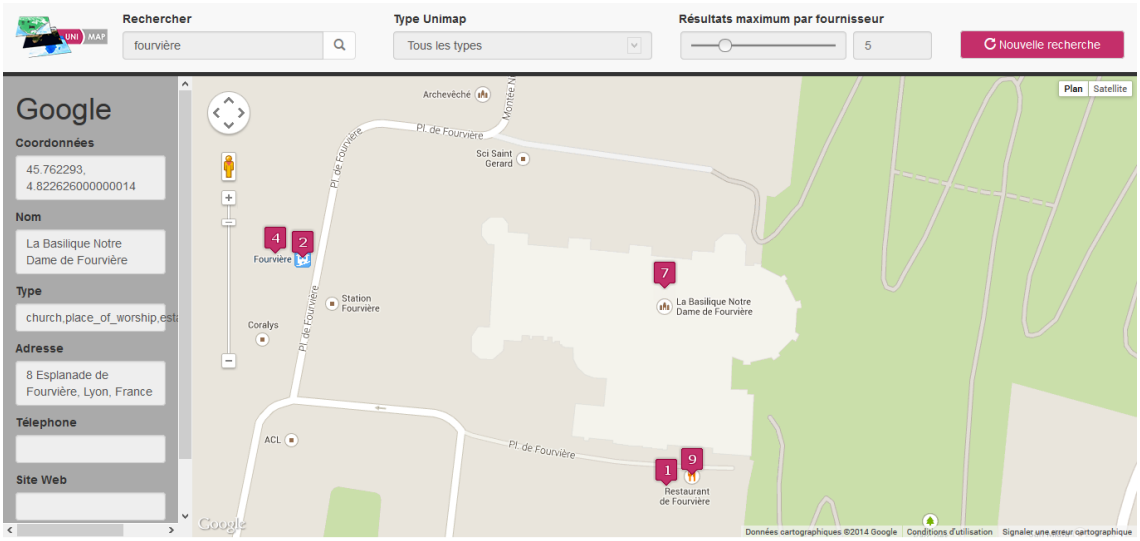

FIGURE 8 – Phase de séléction d'une entité

Elle valide ses paramètres d'entrée, et l'outil interroge les deux autres fournisseurs disponibles, Here et Geonames. Après quelques instants, l'outil lui affiche le fond de carte Here avec 5 entités, et le fond de carte Geonames avec 5 entités également. Chacune des 10 entités proposées forme une correspondance potentielle quand on l'associe avec l'entité Google Maps. Emma a donc 10 correspondances potentielles à vérifier et (in)valider. Quand elle clique sur l'une des entités proposées, les données relatives à celle-ci sont affichées pour lui permettre

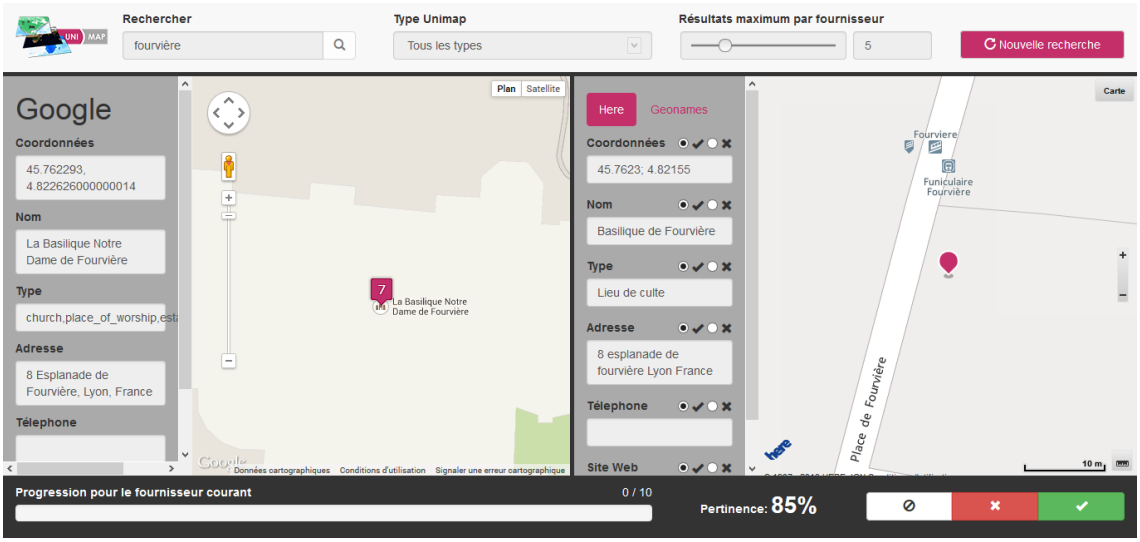

FIGURE 9 – Phase d'alignement

d'expertiser avec un maximum d'information. Emma sélectionne la première entité proposée sur Here, identifiée par « 250u05km-5e58140d8ea140ed8cb5956d68a67f8b ».

En analysant les informations de cette entité, elle s'aperçoit que l'entité représente le POI « Basilique de Fourvière ». Il s'agit bien de l'entité correspondante attendue pour l'entité Google Maps. Emma indique donc que la correspondance potentielle entre l'entité Here « 5e58140d8ea140ed8cb5956d68a67f8b » et l'entité Google Maps « c128fe99cb266174f8a50b94c99cbc13a7f58375 » est correcte, puis elle vérifie et/ou corrige les différences entre les deux entités : hétérogénéité des valeurs (nom, type), éloignement géographique, etc. Lorsqu'elle valide les informations, l'outil enregistre la saisie. Emma peut continuer son expertise en sélectionnant une autre entité proposée par l'outil.

C Annexe 3

Cette annexe inclut notre algorithme d'alignement, qui est détaillé en Section (4.2)

Algorithm 2 Matching algorithm

Require: Une entite souce e
Require: Des ensembles d'entités $\mathcal{C}_1$ ... $\mathcal{C}_n$
Output : Des correspondances entre l'entité e et les entités des fourniseurs

function matching(e)
for all $C_i \in C_1 ... C_n$ **do**
  for all $e_{iz} \in C_i$ **do**
    $latitudeSourceRad \leftarrow 180 * latitude_e / \Pi$
    $longitudeSourceRad \leftarrow 180 * longitude_e / \Pi$
    $latitudeCibleRad \leftarrow 180 * latitude_{e_{iz}} / \Pi$
    $longitudeCibleRad \leftarrow 180 * longitude_{e_{iz}} / \Pi$
    $distance \leftarrow \arccos(\sin(latitudeSourceRad) * \sin(latitudeCibleRad) + \cos(latitudeSourceRad) * \cos(latitudeCibleRad) * \cos(longitudeCibleRad - longitudeSourceRad)) * rayonDeLaTerre$
    $seuilDistance \leftarrow getSeuilDistance(type_{e_{iz}})$
    if $distance < seuilDistance$ **then**
      $scoreDistance \leftarrow 0$
    else
      $scoreDistance \leftarrow 1 - (distance / seuilDistance)$
    end if
    $distanceLevenshtein \leftarrow Levenshtein(label_e, label_{e_{iz}})$
    $longMax \leftarrow max(nom_e, nom_{e_{iz}})$
    $scoreLabel \leftarrow 1 - (distanceLevenshtein / longMax)$
    $scoreType \leftarrow getScoreType(type_e, type_{e_{iz}})$
    $adresseSource \leftarrow getFormatAdresse(adresse_e)$
    $adresseCible \leftarrow getFormatAdresse(adresse_{e_{iz}})$
    $distanceLevenshtein \leftarrow Levenshtein(adresseSource, adresseCible)$
    $longMax \leftarrow max(adresse_e, adresse_{e_{iz}})$
    $ScoreAdresse \leftarrow 1 - (distanceLevenshtein / longMax)$
    $distanceLevenshtein \leftarrow Levenshtein(tel_e, tel_{e_{iz}})$
    $longMax \leftarrow max(tel_e, tel_{e_{iz}})$
    $ScoreTel \leftarrow 1 - (distanceLevenshtein / longMax)$
    $distanceLevenshtein \leftarrow Levenshtein(web_e, web_{e_{iz}})$
    $longMax \leftarrow max(web_e, web_{e_{iz}})$
    $ScoreWeb \leftarrow 1 - (distanceLevenshtein / longMax)$
    $scoreGlobal \leftarrow 2/3 * (scoreDistance + scoreLabel + scoreType/3) + 1/3(ScoreAdresse + ScoreTel + ScoreWeb/3)$
  end for
end for
end function