INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES STRASBOURG

Projet de Recherche Technologique : Développement d'une approche basée sur l'apprentissage automatique pour la similarité entre contradictions

Nathan Schmitt et Léon Marteau

Tuteurs : Ahmed Samet et Guillaume Guarino

Recherches bibliographiques

1 Problématique du sujet

Le but de notre PRT est de proposer à un utilisateur formulant un problème une série de brevets apportant des solutions à des problèmes similaires au sens de la méthode TRIZ : l'utilisateur formule une contradiction et notre programme cherche des brevets avec une contradiction similaire dans une base de donnée.

La méthode TRIZ [1] est une méthode d'innovation systématique qui se base sur le principe des contradictions : on souhaite améliorer une caractéristique d'un système, mais cela provoque des dégradations sur d'autres caractéristiques.
La théorie TRIZ répertorie 39 paramètres et les organise dans une matrice : suivant quel paramètre est à améliorer et quel paramètre est dégradé, un certain nombre de principes sont proposés afin de résoudre ce dilemme.

Dans le cadre de sa thèse, M. Guarino a créé une base de donnée de brevets et a entraîné une intelligence artificielle pour identifier une contradiction au sens de TRIZ dans chaque brevet et en identifier les paramètres.
Nous nous basons sur cette base de données pour entraîner un autre modèle : à partir d'une phrase de contradiction donnée par l'utilisateur, le système doit proposer un certain nombre de brevets apportant des solutions concrètes à des contradictions similaires.

2 État de l'art

Avec en entrée une contradiction formulée par l'utilisateur et les contradictions trouvées dans les brevets, notre principal objectif sera d'identifier lesquelles ont un sens similaires. Cette branche de l'intelligence artificielle, chargée de comprendre la sémantique des phrases, s'appelle le Natural Language Processing (NLP). On s'intéressera plus spécifiquement à l'analyse de similarité entre textes.[2] [3]

2.1 Embeddings

Afin de pouvoir traiter des phrases avec les outils mathématiques habituels, il faut que chaque mot ou phrase soit représenté numériquement. Les embeddings de mots répondent à ce besoin en associant à chaque mot un vecteur d'une dimension N, identique pour tous les mots. Les mots de sens proches, ou pouvant être échangés dans une phrase vont avoir une distance faible dans l'espace de dimension N. La connaissance des embeddings de deux mots permet de savoir s'ils sont sémantiquement proches mais ne permet pas de savoir quel est leur sens.

Différents embeddings existent déjà et ont été calculés sur de grands corpus de textes. Ils permettent d'avoir les embeddings d'un mot simplement en parcourant une table associant mot et embedding. C'est le cas notamment de word2vec [4] de Google, GloVe [5] ou fastText

[6] de Facebook.
Chaque mot est associé à un unique embedding, peu importe le contexte. Cela risque de poser des problèmes dès lors que l'on a affaire à des homonymes.

Depuis 2018, BERT [7], développé par Google, permet de calculer des embeddings de mots dans leur contexte. Dans le cas des brevets, qui portent sur des sujets techniques, certains mot changes radicalement de sens par rapport à leur signification habituelle. Une "solution" chimique n'a aucun lien avec une "solution" de problème. BERT renverra alors deux embeddings très différents pour ce même mot dans deux contextes différents.

BERT ayant fait une avancée majeur et créé un gap de performance avec les méthodes précédentes, de nombreuses variantes ont été développées pour des applications spécifiques. RoBERTa [8] a été entraîné sur plus de données et en changeant les objectifs d'entraînement. DistilBERT [9] et DistilRoBERTa réduisent le nombre de couches cachées pour accélérer le réseau neuronal tout en gardant une performance similaire. PatentBERT[10], PatentSBERTa [11] et Bert for Patent [12] ont été affiné avec des brevets anglophones pour effectuer de la classification automatique de brevets.

D'autres approches d'entrainement très différentes mais gardant le même type de réseau ont aussi été développés. XLNet [13] se place en concurent de BERT en compensant son point faible : le masquage aléatoire des mots durant l'entraînement. D'autres points faibles en émergent et MPNet [14] associe XLNet et BERT pour ne garder que les avantages de chacun.

Le calcul de similarité entre deux phrases n'est pas aisée avec BERT. Si l'on envoie les deux phrases, le temps de calcul pour comparer une requête à une base de donnée est énorme. Si l'on veut pré-calculer les embeddings pour chaque phrase et mesurer ensuite les distancse, le résultat est moins efficaces que GloVe.
SBERT [15] choisi d'optimiser la deuxième option. Il est entraîné avec un réseau siamois. On calcule les embeddings de deux phrases avec le réseau et on étudie leur distance. On cherche à rapprocher les phrases similaires et à éloigner les phrases différentes. Cette méthode est adaptable à tous les embeddings de mots, elle permet de pré-calculer les embeddings de chaque phrase de la base de donnée et d'accélérer par la suite la comparaison de embeddings.

2.2 Distances

Après avoir calculé l'embedding d'un mot ou d'une phrase, on cherche a le comparer d'autres mots ou phrases connues pour déterminer son sens. le principe est simple : puisque les embeddings sont des vecteurs, plus deux vecteurs sont proches, plus les mots correspondants ont des sens proches, il va donc s'agir de calculer la distance entre les deux embeddings qui nous intéressent.

Nous nous intéressons à plusieurs distances différentes [16] :

- cosine : mesure de l'angle séparant deux vecteurs, sans considération de leur norme.
- euclidienne : mesure de la distance séparant deux vecteurs, autrement dit, c'est la norme de leur différence. Cette méthode peut poser des problèmes de normalisation.
- Distance de Manhattan : mesure correspondant à la norme 1, c'est la somme des valeurs absolues des différences de chaque coordonnée des vecteurs :

$$D_m = |x_1 - y_1| + \cdots + |x_N - y_N|$$

- Word Mover Distance [17] : Cette méthode est utilisée pour comparer des documents. Elle crée pour chaque document un vecteur qui recense pour chaque mot sa fréquence d'apparition dans le document. Elle crée ensuite une matrice mettant en relation ces deux vecteurs : la matrice de flux indique combien de fois chaque mot du premier document peut etre relié à chaque mot du second. Elle calcule ensuite différents coefficients a partir de cette matrice pour déterminer la proximité entre les deux documents.

2.3 Autres outils : stemming et lemmatisation

Le stemming et la lemmatisation sont deux outils similaires, nous allons essayer de montrer leur utilité dans le cadre du NLP et d'expliquer la différence entre les deux.
Le stemming et la lemmatisation sont deux outils qui ont pour but d'atteindre la racine d'un mot afin d'étendre sa compréhension :
Le stemming [18] [19] (racinisation en français) consiste a retirer tous les préfixes et suffixes afin de ne garder que son radical, la racine du mot, même si celle-ci ne constitue pas un mot valide dans la langue étudiée. Par exemple, courbe, courbure, courber et courbature ont tous pour racine "courb".
La lemmatisation [20] consiste à projeter un mot sur sa forme la plus simple : l'infinitif, pour une verbe par exemple. Ce lemme est toujours un mot valide dans la langue.

Ces méthodes permettent de simplifier la compréhension du langage, par exemple dans le cadre d'une recherche internet, on pourra trouver des poissonneries à partir de la recherche "poisson".

3 Identification des outils adaptés

Le domaine des brevets est très spécialisé et de nombreux termes ne sont pas utilisés dans leur signification habituelle. On utilisera donc des embeddings qui tiennent compte du contexte : BERT et ses variantes ainsi que XLNet et MPNet.
Afin de comparer rapidement l'entrée de l'utilisateur à toute la base de donnée des brevets, on utilisera les travaux sur SBERT pour pré-calculer les embeddings de phrases.

On ignorera le stemming car les embeddings retenus tiennent compte du contexte dans la phrase et donc simplifier les mots n'aura pas d'impact positif sur le traitement de l'embedding.

Références

[1] Wikipedia : TRIZ. https://fr.wikipedia.org/wiki/TRIZ.

[2] Adrien Sieg for medium : Text Similarities : Estimate the degree of similarity between two texts. https://medium.com/@adriensieg/text-similarities-da019229c894.

[3] Marie Stephen Leo for towards datascience : Semantic Textual Similarity. https://towardsdatascience.com/semantic-textual-similarity-83b3ca4a840e.

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

[5] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove : Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[6] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5 :135–146, 2017.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert : Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv :1810.04805*, 2018.

[8] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta : A robustly optimized bert pretraining approach. *arXiv preprint arXiv :1907.11692*, 2019.

[9] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert : smaller, faster, cheaper and lighter. *arXiv preprint arXiv :1910.01108*, 2019.

[10] Jieh-Sheng Lee and Jieh Hsiang. Patentbert : Patent classification with fine-tuning a pre-trained bert model. *arXiv preprint arXiv :1906.02124*, 2019.

[11] Hamid Bekamiri, Daniel S Hain, and Roman Jurowetzki. Patentsberta : A deep nlp based hybrid model for patent distance and classification using augmented sbert. *arXiv preprint arXiv :2103.11933*, 2021.

[12] Rob Srebrovic and Jay Yonamine. Leveraging the bert algorithm for patents with tensorflow and bigquery, 2020.

[13] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet : Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

[14] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. Mpnet : Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, 33 :16857–16867, 2020.

[15] Nils Reimers and Iryna Gurevych. Sentence-bert : Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv :1908.10084*, 2019.

[16] Manuel Gijón Agudo, Armand Vilalta Arias, and Dario Garcia-Gasulla. Analyzing distances in word embeddings and their relation with seme analysis. In *Artificial Intelligence Research and Development*, pages 407–416. IOS Press, 2019.

[17] TowardAI : Word mover distance. https://towardsai.net/p/nlp/word-movers-distance-wmd-explained-an-effective-method-of-document-classification-89cb258401f4.

[18] Wikipedia : Stemming. https://fr.wikipedia.org/wiki/Racinisation.

[19] Stanford edu : Stemming and lemmatisation. https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html.

[20] Wikipedia : Lemmatisation. https://fr.wikipedia.org/wiki/Lemmatisation.