On the Choice of a Semantic Model

Pierangelo Lombardo · *Intervieweb S.r.l - Zucchetti Group*

KEYWORDS
AI · NLP · Semantic · Word Embedding · Benchmark · Recruiting · HR Tech

INTRODUCTION

Over the last few years it has become more and more evident that Artificial Intelligence is going to deeply affect the way we analyse natural language data. One of the reasons is the progress of techniques called *unsupervised word embeddings*, which take advantage of the semantic information coded in texts to automatically create semantic models. The practical applications of these techniques are countless, and it is quite likely that you will come in touch with some of them in the near future, if you haven't already. If you are curious to gain a little more insight on the subject, then keep reading! You will also find some useful references, in case you want to deepen your knowledge.

As a specific example, let us focus on one of the goals of Inda[1], that needs a semantic model to support HR in some common and repetitive tasks related with recruiting. Recruiters often need to screen the resumes collected in response to a job position or as a result of spontaneous applications. They typically want to select the resumes which are the most relevant for a set of skills or the resumes which are the best fit for a job description. They want therefore a model able to identify the texts (the resumes) which are the best matches for a given set of words (the skills) or the texts which are the best fit for another text of a different kind (the job description). Semantic models (and in particular word embeddings) are instruments which allow to translate into mathematics the concept of "best fit" for texts or words[11–15, 19].

Thanks to the progress of unsupervised word embeddings (and their cousins, the unsupervised document embeddings), we just need to collect a set of text documents of the desired typology (in our case a set of resumes and a set of job descriptions) and use an unsupervised word embedding algorithm to build a semantic model, with no need to invest time and resources to label texts or create ontologies. It might seem that, because of word embeddings, the creation of a semantic model will be easy and almost effortless. Is this so? Not exactly: on the one hand it's true that these algorithms are powerful instruments to tackle a variety of semantic tasks, but, on the other hand, if you want to create a good semantic model, you will have to face many complex choices.

First of all you need to choose the word embedding algorithm, which is a non-trivial choice, as different algorithms perform better in different tasks; after choosing the algorithm, you need to do the so-called *model validation*, i.e., to understand which is the optimal value for each hyperparameter of the model (for instance, one hyperparameter is the number of dimensions in the representation space). Likely, you also need to choose the exact way in which the word embedding is used to solve the semantic tasks you are interested in: for instance if you want to find which resume is the best match for a job description, will you consider the whole text of your resume/job description as equally important or will you focus on some parts of the text? And how do you identify the important parts? As you have many choices to make, you end up with many candidate models, with no *a priori* reason why you should prefer one among the others. So, how can you decide which one you should pick up?

FIVE TIPS TO CHOOSE A SEMANTIC MODEL

In the following, you will find five tips which can help you to deal with the choices described in the previous section.

1. Build an evaluation metric which is suitable for your goal

This may seem obvious, but it's worth saying that if you need to choose among many models, you need a metric which measures the performance of each model, so you will be able to pick up the model with the best performance. The fundamental step to define the evaluation metric is the choice of the semantic task that you want to perform during the evaluation of the models. Once you have chosen the evaluation task, you can probably take advantage of the scientific literature to define the exact mathematical form of your evaluation metric.

2. Evaluate the models while performing the same task you want to solve

The scientific literature of word embedding evaluation confirms the quite self-evident truth that if you want to pick the semantic model which is the best at performing a specific task (the so-called *downstream task*), you should evaluate the candidate models while performing the same task[4, 5, 10, 18]. This downstream task is by definition the optimal evaluation ground to choose your model, this is undeniable. However, sometimes the collection of data to perform the downstream task in a supervised way may be difficult or very expensive, so you may be forced to a tradeoff on the size of the collected data.

3. Be aware of overfitting

Labelled data – such as those collected for the supervised execution of the downstream task – are valuable and limited, and data scientists are always tempted to exploit them beyond the limits enforced by the generalization theory. A likely outcome of this situation is the creation of models which perform extremely well on the dataset used to tune them, but with poor performance on different datasets. This goes under the name of overfitting, and it is one of the biggest problems in machine learning, so you must be extremely careful to avoid (or at least limit) it. One golden rule to prevent overfitting is to avoid the over-exploitation of any single validation dataset[3, 9].

4. Use different evaluation methods

As we said before, in order to avoid overfitting, you should limit the usage of the most valuable datasets, in order to use them only for a few crucial decisions. Therefore, whenever possible, you should base your choices on other datasets. For this reason it's a good practice to collect or create any sort of datasets which may be useful to evaluate your candidate semantic models; an example of simple and re-usable datasets are those designed for the evaluation of the word embedding (the so-called *intrinsic evaluation*), which are quite adaptable and reusable. Even if you already know that the intrinsic evaluation is sub-optimal for the evaluation over your downstream task, it could be extremely useful, in particular if you use many different approaches to evaluate the word embedding (as any approach has different weak points, and together they become more robust).

5. Be aware of the hubness problem

Whenever you perform the intrinsic evaluation of a word embedding, you must be aware of an issue which may seem obscure and yet is very real: the *hubness problem*. Due to the high dimensionality of the embedding space, the majority of words in the embedding tend to have as nearest neighbour one of a small group of favoured words called *hubs*. This is likely to have repercussions on your downstream task and you should therefore take it into account by penalizing the models which are more subscettible to the hubness problem (one way to recognise them is to check the models' performance with rare words)[6–8, 16, 17].

A PRACTICAL EXAMPLE

These five tips have been of great help in the improvement of the semantic model in Inda[1], in recruitment realm, and I think they might be useful also in different contexts, as long as you want to choose/validate a semantic model. If you want to gain a better understanding on the evaluation of a word embedding, I'd suggest you get your hands dirty with a practical example: here[2] you find an app which allows you to play with the evaluation procedure by choosing, among two proposed pairs of words, the one which contains the most semantically similar words. The app is in Italian, but if you don't known the meaning of any word, you can use the integrated web search tool to obtain

a description.

REFERENCES

1. Inda - INtelligent Data Analysis for HR, https://inda.ai
2. Lavaember, https://lavaember.inda.ai
3. Abu-Mostafa, Y.S., Magdon-Ismail, M., Lin, H.T.: Learning from data, vol. 4. AMLBook New York, NY, USA: (2012)
4. Bakarov, A.: A survey of word embeddings evaluation methods. arXiv preprint arXiv:1801.09536 (2018)
5. Blanco, R., Halpin, H., Herzig, D.M., Mika, P., Pound, J., Thompson, H.S., Tran, T.: Repeatable and reliable semantic search evaluation. Journal of web semantics **21**, 14–29 (2013)
6. Dinu, G., Lazaridou, A., Baroni, M.: Improving zero-shot learning by mitigating the hubness problem. arXiv preprint arXiv:1412.6568 (2014)
7. Feldbauer, R., Leodolter, M., Plant, C., Flexer, A.: Fast approximate hubness reduction for large high-dimensional data. In: 2018 IEEE International Conference on Big Knowledge (ICBK). pp. 358–367. IEEE (2018)
8. Francois, D., Wertz, V., Verleysen, M.: The concentration of fractional distances. IEEE Transactions on Knowledge and Data Engineering **19**(7), 873–886 (2007)
9. Goodfellow, I., Bengio, Y., Courville, A.: Deep learning. MIT press (2016)
10. Halpin, H., Herzig, D.M., Mika, P., Blanco, R., Pound, J., Thompon, H., Tran, D.T.: Evaluating ad-hoc object retrieval. In: IWEST@ ISWC (2010)
11. Harris, Z.S.: Distributional structure. Word **10**(2-3), 146–162 (1954)
12. Lai, S., Liu, K., He, S., Zhao, J.: How to generate a good word embedding. IEEE Intelligent Systems **31**(6), 5–14 (2016)
13. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
14. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
15. Mikolov, T., Yih, W.t., Zweig, G.: Linguistic regularities in continuous space word representations. In: Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies. pp. 746–751 (2013)
16. Radovanović, M., Nanopoulos, A., Ivanović, M.: Hubs in space: Popular nearest neighbors in high-dimensional data. Journal of Machine Learning Research **11**(Sep), 2487–2531 (2010)
17. Radovanović, M., Nanopoulos, A., Ivanović, M.: On the existence of obstinate results in vector space models. In: Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval. pp. 186–193 (2010)
18. Rogers, A., Ananthakrishna, S.H., Rumshisky, A.: What's in your embedding, and how it predicts task performance. In: Proceedings of the 27th International Conference on Computational Linguistics. pp. 2690–2703 (2018)
19. Turian, J., Ratinov, L., Bengio, Y.: Word representations: a simple and general method for semi-supervised learning. In: Proceedings of the 48th annual meeting of the association for computational linguistics. pp. 384–394. Association for Computational Linguistics (2010)