French official jobs' taxonomy thanks to NLP *
Machine Learning for Natural Language Processing 2021

Simon Demouchy
ENSAE
simon.demouchy@ensae.fr

Quentin Deltour
ENSAE
quentin.deltour@ensae.fr

Abstract

In this project, we have trained several NLP models aiming to automatically classify jobs into categories according to their names. [1] [2]

Dataset

We worked on the database from Pôle Emploi ROME classification of 11000 different jobs (available here). In this dataset, every job is associated with a "Grand Domaine", a "Domaine" and a "fiche Rome".

For instance, the job "Médecin immunilogue" has the following attributes:

- Fiche Rome : Médecine généraliste et spécialisée
- Domaine : Praticiens médicaux
- Grand domaine : Santé

In the dataset, we have 14 "Grand domaines", 110 "Domaines" and 532 "Fiches Rome". We will try to build several models to classify with these classes, but we know that it will be easier to make a classification on the "Grand domaines" than on the "Fiches Rome".

1 Classifying Jobs

In this section, we will present the different supervised methods that were used in order to classify the jobs titles onto the classification described above.

For all these models, our inputs are the short sentence describing each jobs, and the classes are Grand Domaine, Domaine and Code ROME.

[1]Colab : https://colab.research.google.com/drive/19TKsL6pX-5YGTFoODSRCTjZnX98R_Ion

[2]Github : https://github.com/simondemouchy/NLP_Project

1.1 Baseline model

To have an idea of the accuracy that would obtain a very simple model for this task, we used a Random Forest on the job's title transformed onto a Tfidf matrix.

1.2 Classifiying jobs with CBOW Model

Then we used a CBOW [3] modelization. The architecture of the CBOW (continious-bag-of-words) model is quiet straightforward. Firstly we build an embedding layer mapping each word to a vector representation. Then, we compute the vector representation of all words in each sequence and average them. Finally, we add a dense layer to output the number of classes and the softmax function in order to get a probability on every class.

1.3 Classifiying jobs with Transformer

Because we worked on french official jobs, we have worked with CamemBert [2], a state-of-the-art language model for doing NLP with text written in french. We had to adapt the CamemBert model to our specific task of short sentence classification. Therefore we have created our own model, which is the addition of the CamemBert model and a dense layer to carry out the job classification task. We also add a dropout to regularize the model.

1.4 Classifiying jobs with Fast text

As we perform a simple sentence classification task, we wanted to use the fast text models [1].

Fast Text is especially good at short sentence classification as it uses character level information: each sentence is split at the same time onto ngrams of words and ngrams of **characters**.

We used the autotune specification from Fast Text supervised training in order to find the best hyperparameters for our final model. We are interested in the hyperparameters specifying the number of characters (respectively words) for the ngrams of characters (words).

As a result, we find that the best hyperparameters are the following:

- **maxn** (max length character Ngrams): 5
- **minn** (min length character Ngrams): 2
- **word Ngrams**: 5

2 Comparison of results

The accuracy obtained from all these models are described in the table 1.

Table 1: Comparison of accuracy of the models

	Grand Domaine	Domaine	ROME
Baseline	68.45	60.97	53.11
CBOW	89.92	86.29	83.70
CamemBert	91.23	83.17	29.72
Fast Text	91.97	87.00	81.86

All our models (except baseline) are close when comparing accuracy. However, we can say that the fast text is the best for classifying Grand Domaine, but looses more accuracy when switching to more classes, and CBOW becomes better for ROME classification.

We can also plot a confusion matrix to understand where the mistakes are made. Figure 1 in appendix present the confusion matrix of the classification of grand domaine by the fast text model.

Most classes do not have much errors, and classes with more observations (Industry and Services) are the one with the most errors.

3 Assessing the models' external validity on another Dataset

Then, we desired to validate the models on another jobs'name. Therefore, we used a dataset of annotated jobs from ONISEP (French public institute of information on education and jobs).

We get the following results for the prediction of "grand domaine":

- CBOW model : Accuracy of 73%;
- CamemBert model : Accuracy of 78%;
- Fast Text model : Accuracy of 70%.

It seems that the Camembert model is better for the extrapolation to external datasets, while the Fast Text looses a lot in accuracy (nearly 20pp).

4 Checking the robustness to spelling mistakes

In order to assess the validity of our models, we wanted to see how it would behave when spelling mistakes are done in the name of the jobs (we took the case of one letter forgotten). This can often be the case in surveys, where users will fill their job and can be subject to spelling mistakes or inversion of characters.

Figure 2 present the evolution of the accuracy of the different models when we increase the number of job's title that has a spelling mistakes in it.

Transformers model still has the best performance of the 3 models. However, it seems that the fast text models (thanks to the character level information) is not too much subject to spelling mistakes, and shows a relatively small decrease in its accuracy.

5 Conclusion

We have shown that this task of jobs classification was possible to execute thanks to NLP models. The accuracy we obtain on Pôle Emploi dataset is good, but it has a limited external validity as we have seen as we loose 10 to 20 pp in accuracy when looking at job's title that are not in the training dataset. However, it is still a convincing result (more than 70% accuracy), and it is robust to spelling mistakes (when including character level information such as Fast Text), which makes it a powerful tool for studying surveys on which people are asked to fill their profession in a blank space.

To further this analysis, I think it would have been interesting with more time to enlarge the training dataset with spelling mistakes (the way we did with dropping one letter, or using Levenshtein distance to create job's titles that are really close from the one we have but with one letter substitution for example). To continue in our idea of being efficient for treating surveys, we could use surveys from INSEE (for example Enquete Emploi) provided their jobs are classified in a way to retrieve the grand domaine classification.

References

[1] Armand Joulin et al. “Bag of Tricks for Efficient Text Classification”. In: *arXiv preprint arXiv:1607.01759* (2016).

[2] Louis Martin et al. “CamemBERT: a Tasty French Language Model”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020, pp. 7203–7219.

[3] Tomas Mikolov et al. “Efficient Estimation of Word Representations in Vector Space”. In: *arXiv preprint arXiv:1301.3781* (2013).

A Appendix

Figure 1: Confusion Matrix of Grand Domaine predicted by fast text

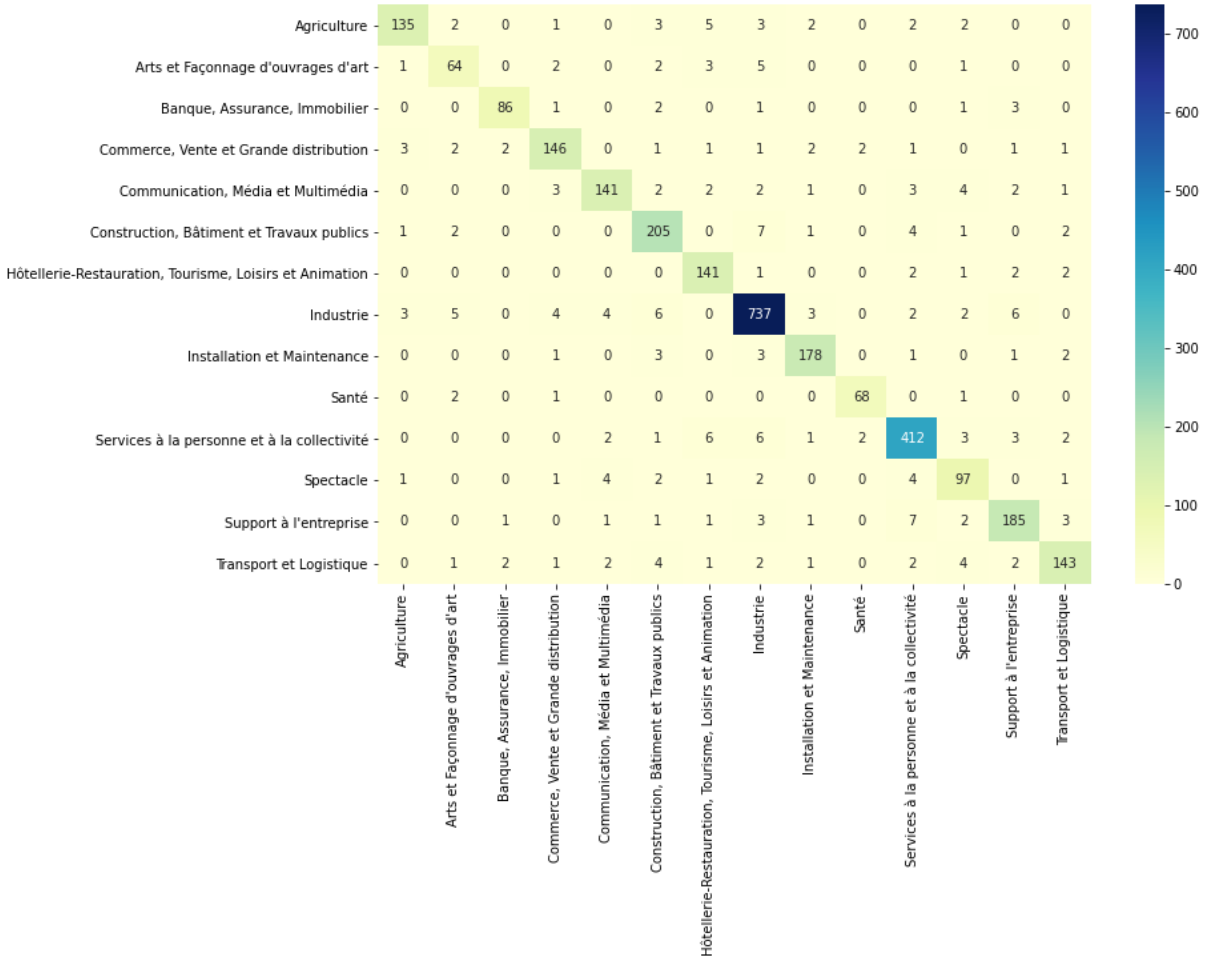

Figure 2: Evolution of accuracy of different models when spelling mistakes are introduced