Resume: A Robust Framework for Professional Profile Learning & Evaluation

Clara Gainon de Forsan de Gabriac
clara.gainon_de_forsan_de_gabriac@lip6.fr
Sorbonne Université, CNRS, LIP6
Paris, France

Amina Djelloul
amina.djelloul@etu.upmc.fr
Sorbonne Université, CNRS, LIP6
Paris, France

Constance Scherer
constance.scherer@etu.upmc.fr
Sorbonne Université, CNRS, LIP6
Paris, France

Vincent Guigue
vincent.guigue@lip6.fr
Sorbonne Université, CNRS, LIP6
Paris, France

Patrick Gallinari
patrick.gallinari@lip6.fr
Sorbonne Université, CNRS, LIP6
Paris, France

CCS CONCEPTS

• **Computing methodologies** → **Natural language generation**; **Information extraction**; **Semi-supervised learning settings**; **Neural networks**.

KEYWORDS

deep learning, neural networks, professional profile extraction, semi-supervised learning

This work was published at the ESANN 2020 Conference

Introduction. Learning to match candidates with job offers is a major challenge for any institution's human-resources department. The fast development of online job-boards (*Monster*, *JobTeaser*…) and professional social networks such as LinkedIn, makes this task increasingly crucial. As such, improving profile modeling on both users and jobs may allow developing new tools to suggest relevant skills and to connect unformatted job titles and descriptions to standardized ontologies like ESCO (*https://ec.europa.eu/social/main.jsp?catId=1326*).

We propose a method to learn and evaluate professional profiles using the information contained in a user's LinkedIn profile. Unlike traditional Expertise Matching methods that mainly rely on categorical data, we aim at building meaningful professional representations using only user-generated texts (their job titles and descriptions) during training in a self-supervised setting. We want our profiles to encode a sufficient amount of information to predict the future of users' careers. We also try to determine the skills and the industrial field associated with a profile.

While text representation has long been performed at the document level in a bag-of-words setting [2], Word2vec [7] enables us to predict words in a local context opening the way for meaningful word embeddings and text generation applications. A second generation of language models introduces a solution to take into account out-of-vocabulary words through subword information encoding [3] and even more recent proposals focus on contextual embeddings and generative settings to improve sentence understanding [9].

In the field of professional profile extraction, [1] exploit a supervised framework to identify fake profiles on LinkedIn; our approach is closer to Text Summarization [6, 8] in the sense that we do not rely on supervision to learn profiles. On top of that, we also aim at generating texts to predict the next job of a particular user. Our framework *Resume* relies on state-of-the-art robust language models to encode textual information [3, 9]. Then we aggregate job embeddings to build a user representation. On top of this framework, our main contribution resides in the evaluation approach; we were provided with more than 500k LinkedIn pages (Fig. ??) which enables us to measure quantitatively our ability to predict skills and industrial field for our set of users. We also provide an original RNN based generative approach for the next job prediction task as well as an evaluation procedure relying on a summarizing metric [5]. In this extended abstract, we introduce our models and learning setups as well as our results, highlighting that the noise level in the raw data leads to a surprising ranking of our approaches.

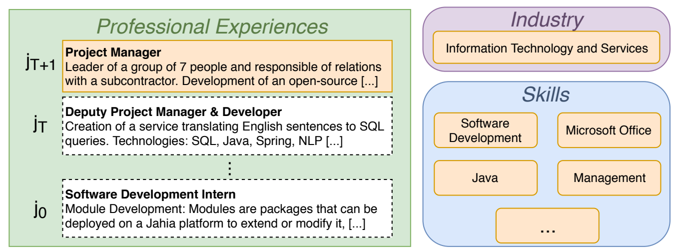

Figure 1: User's profile schema. ***A user is composed of their past jobs, their industry and their skills. The last job, j_{T+1}, the industry and the skills are the labels we use to evaluate our z_u. The skills and industry are categorical values, whereas j_t is free text.***

Models. We aim at building a representation $\mathbf{z}_u$ of our users using only their past jobs. We evaluate the meaningfulness of our users' representations through 3 tasks: the prediction of their skill set, the prediction of their industrial field and the generation of their last job. We refer to them as the Skill Predictor, the Industry Predictor, and the Last Job Decoder. Each of them is fed our user representation and outputs respectively their skill set, the domain they work in (or industry), and their last job. Our approach relies on the fact that a semantically rich representation of our users would allow very simple predictors to extract relevant, implicit information from them.

Table 1: Experimental results on the classification tasks.

Model	F1 score - Skills Prediction	Accuracy - Industry Prediction
Most Common	24.0%	6.3%
FT_{pt} (pre-trained FastText)	40.9%	35.6%
FT_{CV} (CV-oriented FastText)	**42.4%**	**38.4%**
ELMo	39.0%	30.7%

Table 2: Experimental results on job prediction.

Model	Bleu score (Last Job)				
	Bleu	Bleu-1	Bleu-2	Bleu-3	Bleu-4
Most Common	0.00	**33.3**	0.3	0.0	0.0
FT_{pt} (pre-trained FastText)	1.91	20.6	3.5	0.8	0.2
FT_{CV} (CV-oriented FastText)	**2.15**	22.2	**3.8**	**0.9**	**0.3**
ELMo	1.74	22.5	**3.8**	0.6	0.2

Formally, each user's raw data consists in a set $\mathcal{J}_u$ of chronologically ordered free texts: $\mathcal{J}_u = \{j_0, \ldots, j_T, j_{T+1}\}$. The user is also associated to a set of skills described as a binary vector in the skill domain: $\mathbf{s}_u \in \{0,1\}^S$. The industrial field is denoted $b_u \in \{1, \ldots, B\}$. Our models are composed of a job encoder that deals with raw texts and a job aggregator that outputs a user profile: $\mathbf{z}_{j_t} = enc(j_t)$, $\mathbf{z}_u = agg(\mathbf{z}_{j_0}, \ldots, \mathbf{z}_{j_T})$. Then we train three independent components tackling the next job prediction task $\widehat{j}_{T+1} = dec(\mathbf{z}_u)$, the skill prediction $\widehat{\mathbf{s}}_u = f_s(\mathbf{z}_u)$ and the industrial field categorization $\widehat{b}_u = f_b(\mathbf{z}_u)$. Note that while both the Skill Predictor and the Industry Predictor are simple, 2-layers MLP, the Last Job Decoder is composed of an extra recurrent layer so as to generate sentences. Our data is noisy and contains a lot of out-of-vocabulary or rare tokens (*e.g.* product names or misspelled words). We thus choose recent text-encoding models capable of leveraging subwords information: FastText and ELMo. As FastText is light and easy to train, we will compare pre-trained and specifically trained embeddings on our different tasks. ELMo is a recent language model relying on contextual embeddings: words' representations depend on their contexts. In practice, $\mathbf{z}_w$ are obtained after running a bi-directional recurrent neural network over the text: $\mathbf{z}_w$ is an aggregation of the representations of previous words until w in one direction and of the following words in the other direction. As opposed to FastText, ELMo relies upon millions of parameters and we will only use the pre-trained version of this language model. In this work, a job $j_t = (w_1^{(t)}, ..., w_N^{(t)})$ is simply encoded by averaging all its words' representations: $\mathbf{z}_{j_t} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{z}_{w_n^{(t)}}$. Then, we represent users as an aggregation of their jobs: $\mathbf{z}_u = \frac{1}{T+1}\sum_{t=0}^{T} \mathbf{z}_{j_t}$.

Results. The results of our classification tasks on skills and industry prediction and those of text generation are reported in Table 1 and Table 2 respectively. Those results show that the FT_{CV} outperforms the other models in both skills and industry prediction, as well as in text generation. This ranking among our approaches is surprising, it points out that the CV style does not follow a classical language model: a (very) robust and dedicated model is required to tackle misspellings, abbreviations, ellipses, acronyms that characterize the fast writing style observed on CV. The same kind of conclusions has been drawn in [4].

Such results, while initially counter-intuitive, can be empirically understood when taking a closer look at the predictions. The last job generation highlights the difficulty for our models to generate long job descriptions as well as very specific sentences. For instance, the encoding-decoding process gives:

GT: *Title: E-commerce Consultant, Desc: My mission consists in reaching the goals set up by the clients regarding their profitability and/or notoriety issues [...]*

FT_{pt} *Title: Marketing Manager, Desc: Management of the client relationship, Social networks management, Social networks management [...]*

FT_{CV} *Title: Marketing Manager, Desc: Managing the communication strategy and the communication strategy for clients[...]*

ELMo *Title: Secteur Manager, Desc: Management of the client relationship, [UNK], stock management, stock management [...]*

Such predictions highlight the complexity and diversity of our data. While not human-like, those predictions can add a lot of value to a CV database as they capture the essence of a career. The analysis of both skills and industry prediction for all three models indicate that a consequent part of the wrong predictions make sense to a human reader. For instance, a profile containing the skills *Office Pack*, *Photoshop* and *Marketing* is predicted to have the *Microsoft Word*, *Adobe Photoshop* and *Digital Marketing* skills. Similarly, a Developer working in the Pharmaceutical Industry can be either predicted in the "IT" or the "Pharmaceutical" industry. Those observations lead us to believe that the representation of our users is rather satisfactory. Most prediction errors are understandable and could be tackled by a more thorough data pre-processing.

REFERENCES

[1] Shalinda Adikari and Kaushik Dutta. 2014. Identifying Fake Profiles in LinkedIn. In *PACIS*. 278.

[2] David M. Blei, Andrew Y. Ng, Michael I. Jordan, and John Lafferty. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research* 3 (2003), 2003.

[3] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *CoRR* 1607.04606 (2016). arXiv:1607.04606 http://arxiv.org/abs/1607.04606

[4] Pallavi Jain, Robert Ross, and Bianca Schoen-Phelan. 2019. Estimating Distributed Representation Performance in Disaster-Related Social Media Classification. In *ASONAM*.

[5] T. Ward K. Papineni S. Roukos and W. Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *ACL*.

[6] Inderjeet Mani. 1999. *Advances in Automatic Text Summarization*. MIT Press, Cambridge, MA, USA.

[7] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. *CoRR* 1310.4546 (2013). arXiv:1310.4546 http://arxiv.org/abs/1310.4546

[8] R. Paulus, C. Xiong, and R. Socher. 2017. A Deep Reinforced Model for Abstractive Summarization. *CoRR* 1705.04304 (2017). arXiv:1705.04304 http://arxiv.org/abs/1705.04304

[9] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR* 1802.05365 (2018). arXiv:1802.05365 http://arxiv.org/abs/1802.05365