conSultantBERT: Fine-tuned Siamese Sentence-BERT for Matching Jobs and Job Seekers

Dor Lavi
dor.lavi@randstadgroep.nl
Randstad Groep Nederland
Diemen, The Netherlands

Volodymyr Medentsiy
volodymyr.medentsiy@randstadgroep.nl
Randstad Groep Nederland
Diemen, The Netherlands

David Graus
david.graus@randstadgroep.nl
Randstad Groep Nederland
Diemen, The Netherlands

ABSTRACT

In this paper we focus on constructing useful embeddings of textual information in vacancies and resumes, which we aim to incorporate as features into job to job seeker matching models alongside other features. We explain our task where noisy data from parsed resumes, heterogeneous nature of the different sources of data, and crosslinguality and multilinguality present domain-specific challenges.

We address these challenges by fine-tuning a Siamese Sentence-BERT (SBERT) model, which we call conSultantBERT, using a large-scale, real-world, and high quality dataset of over 270,000 resume-vacancy pairs labeled by our staffing consultants. We show how our fine-tuned model significantly outperforms unsupervised and supervised baselines that rely on TF-IDF-weighted feature vectors and BERT embeddings. In addition, we find our model successfully matches cross-lingual and multilingual textual content.

CCS CONCEPTS

• **Computing methodologies** → **Ranking**; • **Information systems** → **Content analysis and feature selection**; **Similarity measures**; **Language models**.

KEYWORDS

job matching, BERT, fine-tuning

1 INTRODUCTION

Randstad is the global leader in the HR services industry. We support people and organizations in realizing their true potential by combining the power of today's technology with our passion for people. In 2020, we helped more than two million job seekers find a meaningful job with our 236,100 clients. Randstad is active in 38 markets around the world and has top-three positions in almost half of these. In 2020, Randstad had on average 34,680 corporate employees and generated revenue of € 20.7 billion.

Each day, at Randstad, we employ industry-scale recommender systems to recommend thousands of job seekers to our clients, and the other way around; vacancies to job seekers. Our job recommender system is based on a heterogeneous collection of input data: curriculum vitaes (resumes) of job seekers, vacancy texts (job descriptions), and structured data (e.g., the location of a job seeker or vacancy). The goal of our system is to recommend the best job seekers to each open vacancy. In this paper we explore methods for

constructing useful embeddings of textual information in vacancies and resumes.

The main requirements of the model are (i) to be able to operate on multiple languages at the same time, and (ii) to be used to efficiently compare a vacancy with a large dataset of available resumes.

Our end-goal is to incorporate these embeddings, or features derived from them, in a larger recommender system that combines a heterogeneous feature set, spanning, e.g., categorical features and real-valued features.

1.1 Problem setting

Several challenges arise when matching jobs to job seekers through textual resume and vacancy data.

First, the data we work with is inherently noisy. On the one hand, resumes are user-generated data, usually (but not always) in PDF format. It goes without saying that parsing those files to plain text can be a challenge in itself and therefore out of scope for this paper. On the other hand, vacancies are usually structured formatted text.

Second, the nature of the data differs. Most NLP research in text similarity is based on the assumption that two pieces of information are the same but written differently [3]. However, in our case the two documents do not express the same information, but complement each other like pieces of a puzzle. Our goal is to match two complementary pieces of textual information, that may not exhibit direct overlap/similarity.

Third, as a multinational corporate that operates all across the globe, developing separate models for each market and language does not scale. Therefore, a desired property of our system is multilinguality; a system that will support as many languages as possible. In addition, as it is common to have English resumes in non-English countries (e.g., in the Dutch market around 10% of resumes are in English), cross-linguality is another desired property, e.g., being able to match English resumes to Dutch vacancies.

This paper is structured as follows. First, in Section 2 we summarize related work on the use of neural embeddings of textual information in the recruimtent/HR domain. Next, in Section 3.1 we describe how we leverage our internal history of job seeker placements to create a labeled resume-vacancy pairs dataset. Then, in Section 3.2 we describe how we fine-tune a multilingual BERT with bi encoder structure [12] over this dataset, by adding a cosine similarity log loss layer. Finally, in Section 4 we describe how using the mentioned architecture helps us overcome most of the challenges described above, and how it enables us to build a maintainable and scalable pipeline to match resumes and vacancies.

2 RELATED WORK

Neural embeddings are widely used for content-based retrieval, and embedding models became an essential component in the modern recommender system pipelines [8] [6]. So the the main focus of our work is to construct embeddings of textual information in vacancy and resume, which could be then incorporated into another job-job seeker matching model and used along with other features, e.g. location and other categorical features. So we will focus on reviewing methods of embedding vacancies and resumes. The task of embedding resume and vacancy could be posed as creating domain-specific document embeddings. Although context-aware embeddings proved to outperform bag-of-words approaches in most of the NLP tasks in academia, the latter is still widely used in the industry.

Bian et al. propose to construct two sub-models with a co-teaching mechanism to combine predictions of those models. The first sub-model encodes relational information of resume and vacancy, and the second sub-model, which is related to our work, encodes textual information in resume and vacancy. Documents are processed per sentence, with every sentence being encoded using the CLS token of the BERT model. The Hierarchical Transformer is applied on top of sentence embeddings to get the document embeddings. The final match prediction is obtained by applying a fully-connected layer with sigmoid activation on concatenated embeddings of resume and vacancy. Bhatia et al. propose to fine-tune BERT on the sequence pair classification task to predict whether two job experiences belong to one person or not. The proposed method does not require a dataset of labeled resume-vacancy pairs. The fine-tuned model is used to embed both the job description of the vacancy and the current job experiences of the job seeker. Zhao et al. process words in resumes and vacancies using word2vec embeddings and domain-specific vocabulary. Word embeddings are fitted into a stack of convolutional blocks of different kernel sizes, on top of which Zhao et al. apply attention to get the context vector and project that into the embedding space using the FC layer. The model is trained using the binary cross-entropy loss on the task of predicting a match between job seeker and vacancy. Ramanath et al. use supervised and unsupervised embeddings for their ranking model to recommend candidates to recruiter queries. The unsupervised method does not use the unstructured textual data but relies on the data stored in Linkedin Economic Graph [15], which represents entities such as skills, educational institutions, employers, employees, and relations among them. Candidates and queries are embedded using the graph neural models in this method. The supervised method embeds textual information in recruiters' queries and candidates' profiles using the DSSM [7] model. The DSSM operates on the character trigrams and is composed of two separate models to embed queries and candidates. The DSSM model is trained on the historical data of recruiters' interaction with candidates.

Zhu et al. utilize skip-gram embeddings of various dimensionalities (64 for resume and 256 for a vacancy) to encode words, which are then passed through two convolutional layers. A pooling operation (max pooling for resume and mean pooling for vacancy) is applied to the output of convolutional layers to get the embeddings of resume and vacancy. The model is optimized using the cosine similarity loss. Qin et al. divide job postings into sections of ability requirements, and resumes into sections of experiences. The words are encoded using pre-trained embeddings and processed with bi-LSTM. To get the embeddings of job postings and resumes, they propose a two-step hierarchical pipeline. Every section is encoded using the attention mechanism, and finally to get embeddings of the job postings and resumes they run bi-LSTM on top of section encodings and aggregate bi-LSTM outputs using the attention mechanism. Additionally, during processing information in resumes Qin et al. propose to add encodings of job postings to emphasize skills in a resume relevant for a specific job.

We can not directly compare our work with other approaches, because of the different datasets used. For example, Bhatia et al. and Qin et al. assume well-structured resumes, which is not the case in our situation. Ramanath et al. builds an embedding model for recruiters' queries which are shorter than vacancies and lack the context provided in the vacancy. Bhatia et al. propose solution when limited amount of data is available. We on the other hand work in a setting of abundant data of heterogeneous nature, but at the same time resumes lack consistent structure, while vacancies are given in a more structured way. Additional issues that are not considered by most of the reviewed works are (i) *cross-linguality*, so that we aim at predicting English resumes to Dutch vacancies if there is a potential match, and (ii) *multilinguality*, where we aim to serve a single model for multiple languages. We address both issues with our approach. Next, Qin et al. and Zhu et al. observe that an embedding model may benefit from constructing parallel pipelines to process resume and vacancy. Our approach relies on a shared embedding model of resumes and vacancies.

While there are many organizations capable to train off the shelf transformers, not many of them have the availability of an abundance of high-quality labeled data. As a global market leader, we are situated in a unique position. We have both rich histories of high-quality interactions between consultants, candidates, and vacancies, in addition to having the content to represent those candidates and vacancies.

3 METHOD

Here we describe our method, more specifically, in Section 3.1 we describe how we acquire our labeled dataset of resume/vacancy-pairs. Next, in Section 3.2 we describe our multilingual SBERT with bi-encoder and cosine similarity score as output layer.

3.1 Dataset creation

We have rich history of interaction between consultants (recruiters) and job seekers (candidates). We define a positive signal any point of contact between a job seeker and consultant (e.g., a phone call, interview, job offer, etc.). Negative signals are defined by job seekers who submit their profile, but get rejected by a consultant without any interaction (i.e., consultant looks at the job seeker's profile, and rejects). In addition, since we have unbalanced dataset, for each vacancy we add random negative samples, which we draw randomly from our job seeker pool. This is done in spirit with other works, which also complement historical data with random negative pairs [2, 17, 18].

Our dataset consists of 274,407 resume-vacancy pairs, out of which 126,679 are positive samples, 109,724 are negative samples
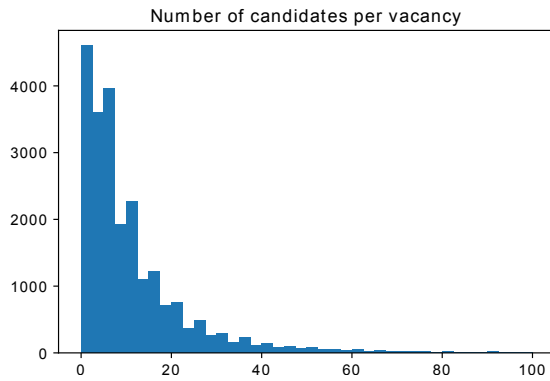
Figure 1: Histogram of the number of resumes per vacancy.

as defined by actual recruiters, and 38,004 are randomly drawn negative samples. We have 156,256 of unique resumes and 23,080 unique vacancy texts, which implies that one vacancy can be paired with multiple resumes. Figure 1 shows the histogram of the number of resume-vacancy samples per vacancy. We see that for the majority of vacancies, we have a small number of paired resumes with approximately 10.5% of our vacancies being paired with a single resume, and approximately 30% of our vacancies being paired with at most three resumes.

resumes are user-generated PDF documents which we parse with Apache Tika.[1] Overall, these parsed resumes can be considered quite noisy input to our model; there is a wide variation in format and structure of resumes, where common challenges include the ordering of different textual blocks, the diversity of their content (e.g., spanning any type of information from personalia, education, work experience, to hobbies), and parsing of tables and bullet points. At the same time, vacancies are usually well structured and standardized documents. They consist of on average 2,100 tokens (while resumes are on average longer and comprised of 2,500 tokens), and are roughly structured according to the following sections: job title, job description, job requirements, including required skills, job benefits, including compensation and other aspects, and company description, which usually includes information about the industry of the job offered in the vacancy.

3.2 Architecture

Our method, dubbed conSultantBERT (as it is fine-tuned using labels provided by our consultants), utilizes the multilingual BERT [5] model pre-trained on Wikipedia pages of 100 languages.[2] We fine-tune it using the Siamese networks, as proposed by Reimers and Gurevych. This method of fine-tuning BERT employs the bi-encoder structure, which is effective in matching vacancy with a large pool of available resumes.

---

[1] https://tika.apache.org/
[2] We used the bert-base-multilingual-cased model from the HuggingFace library [16].

The original formulation of the Sentence-BERT (SBERT) model takes a pair of sentences as input, with words independently encoded in every sentence using BERT, which are aggregated by pooling to get the sentence embeddings. After that, it either optimizes the regression loss, which is MSE loss between the cosine similarity score and true similarity label, or optimizes the classification loss, namely the cross-entropy loss.

Whereas SBERT is aimed at computing pairwise semantic similarity, we show that it can be applied effectively for our task of matching two heterogeneous types of data. Our fine-tuning pipeline is illustrated at the Figure 2. So we pass resume and vacancy pairs to the siamese SBERT and experimented with classification and regression objectives.

3.2.1 Document representation. Most transformer models, including SBERT, aim to model sentences as input, while we are interested in modeling documents.

We experimented with several methods for document representation by using the embeddings of the pre-trained BERT model. First, we attempted to split our input documents into sentences to encode each sentence; we tried different sentence representations, first we used the pooled layer output from BERT to represent each sentence, which we then average to represent the document. Here, we experimented with both simply averaging sentences, and weighted averaging (by sentence length).

Next, we tried to take the mean of the last 4 layers of the <CLS> token, which is a special token placed at the start of each sentence, and considered a suitable representation for the sentence (according to Devlin et al.). With these sentence representations, too, we represented the underlying document through averaging, both weighted and simple averaging.

Our final approach however, was more simple. We ended up treating the first 512 tokens of each input document as input for the SBERT model, ignoring sentence boundaries. To avoid trimming too much content, we pre-processed and cleaned our input documents by, e.g., removing non-textual content that resulted from parsing errors.

3.2.2 Fine-tuning method. We fine-tune our model on our 80% training data split for 5 epochs, using a batch size of 4 pairs, and mean pooling, which was found during hyperparameter tuning on our validation set as optimal parameters. To fine-tune the pre-trained BERT model, we need to trim the content of every resume and vacancy to 512 tokens, as the base model limits the maximum number of tokens to 512.

4 EXPERIMENTAL SETUP

In this section, we describe the final datasets we use for training, validation, and testing in Section 4.1, the baselines and why we employ them in Section 4.2, our proposed fine-tuned conSultantBERT approach in Section 4.3, and finally, in Section 4.4 we explain our evaluation metrics and statistical testing methodology.

4.1 Dataset

As described in Section 3.1, our dataset consists of 274,407 resume-vacancy pairs. We split this dataset into 80% train (219,525 samples), 10% validation and 10% test (27,441 samples each).
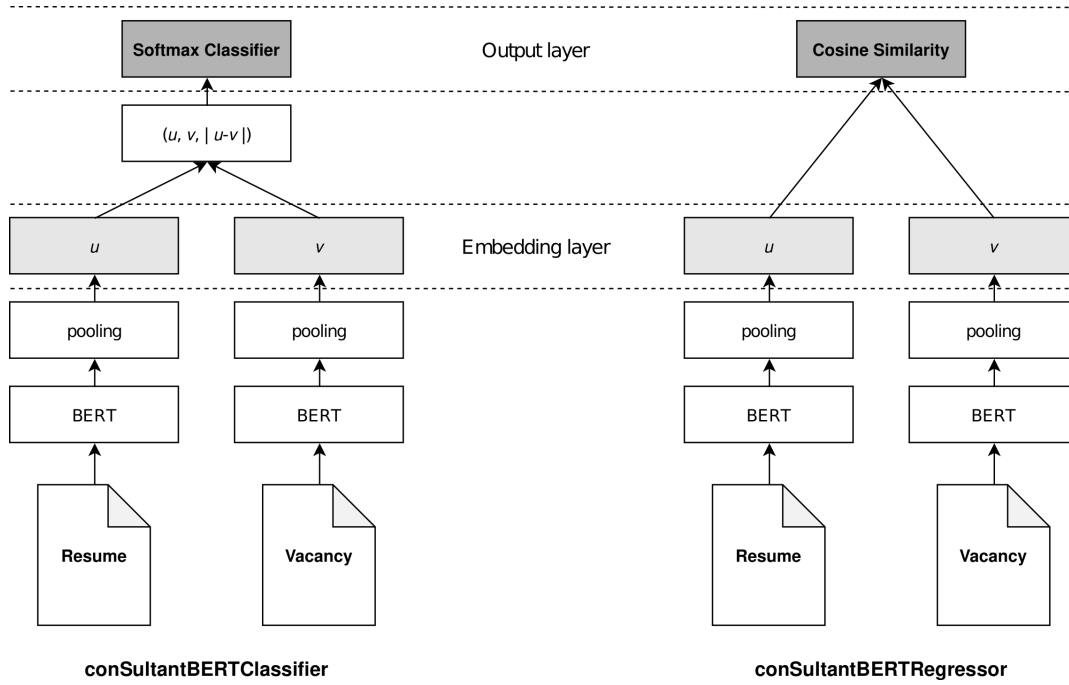
Figure 2: Our conSultantBERT architectures with classification objective (left) and regression objective (right), with resume text input on the left, and vacancy text input on the right-hand side. Image adopted from [12]. Note that vacancy and resume differ at the level of words (vocabulary), but also format, structure, and semantics.

We use our training set to (i) fine-tune the embedding models and train the supervised random forest classifiers, the validation set for the hyperparameter search of the SBERT hyperparams described in Section 3.2.2, and report on performance on our test set.

4.2 Baselines

As this paper revolves around constructing useful feature representations (i.e., embeddings) of the textual information in vacancy and resumes, we compare several approaches of generating these embeddings.

4.2.1 Unsupervised. Our first baselines rely on unsupervised feature representations. More specifically, we represent both our vacancies and resumes as either (i) TF-IDF weighted vectors (TFIDF), or (ii) pre-trained BERT embeddings (BERT).

We then compute cosine similarities between pairs of resumes and vacancies, and consider the cosine similarity as the predicted "matching score." Our TF-IDF vectors have 768 dimensions, which is equal to the dimensionality of BERT embeddings.[3] We fitted our TF-IDF weights on the training set, comprising both vacancy and resume data. As described in Section 3.2, we rely on BERT models pre-trained on Wikipedia from the HuggingFace library [16].

These unsupervised baselines help us to assess the extent to which the vocabulary gap is problematic, i.e., if vacancies and resumes use completely different words, both bag of words-based

approaches such as TF-IDF weighting and embedding representations of these different words will likely show low similarity. Formulated differently, if word overlap or proximity between vacancies and resumes is meaningful, these baselines would be able to perform decently.

4.2.2 Supervised. Next, we present our two supervised baselines, where we employ a random forest classifier that is trained on top of the feature representation described above (TFIDF+RF, BERT+RF). These supervised baselines are trained on our 80% train split, using the default parameters given by the scikit-learn library [9].

We add these supervised methods and compare them to the previously described unsupervised baselines to further establish the extent of the aforementioned vocabulary gap, and the extent in which the heterogeneous nature of the two types of documents mentioned in Section 1.1 plays a role. That is to say, if there is a direct mapping that can be learned from words in one source of data (vacancy or resume), to the other source, supervised baselines should be able to pick this up and outperform the unsupervised baselines.

4.3 conSultantBERT [4]

Finally, we present our fine-tuned embedding model; conSultantBERT, which we fine-tune using both the classification objective as the regression objective, as explained in Section 3.2 and illustrated in Figure 2.

---

[3]We experimented with increasing the dimensions of TF-IDF vectors, but it did not gave any substantial increase in performance (increasing dimensions 3-fold improved ROC-AUC by +0.7%).

[4]The capitalized S in conSultant is a reference to SBERT.

#	Model	ROC-AUC	precision	recall	f1
1.	BERT+Cosine	0.5325	0.5057	0.5006	0.3379
2.	TFIDF+Cosine	0.5730*	0.5890	0.5016	0.3564
3.	BERT+RF	0.6978	0.6421	0.6363	0.6360
4.	TFIDF+RF	0.7174	0.6581	0.6526	0.6527
5.	conSultantBERTClassifier+Cosine	0.7474	0.7001	0.6426	0.5994
6.	conSultantBERTClassifier+RF	0.8366*	0.7642	0.7643	0.7643
7.	conSultantBERTRegressor+Cosine	**0.8459***	0.7714	0.7664	0.7677
8.	conSultantBERTRegressor+RF	0.8389	0.7684	0.7658	0.7666

Table 1: Results of different runs. * denotes statistically significant difference from the alternative in the same group at $\alpha = 0.01$. Best-performing run in bold face.

Our intuition for using the classification objective is our task and dataset; which consists out of binary class labels, making the classification objective the most obvious choice. At the same time, our work revolves around searching for meaningful embeddings, not necessarily solving the end-task in the best possible way, for which, we hypothesize, the model may benefit from the more fine-grained information that is present in the cosine similarity optimization metric.

Similarly to our baselines, we consider (i) our fine-tuned model's direct output layer (i.e., the cosine similarity output layer) as "matching score," between a candidate and a vacancy (conSultantBERT–Regressor+Cosine, conSultantBERTClassifier+Cosine), in addition to the predictions made by a supervised random forest classifier which we train on the embedding layer's output (see Figure 2), yielding the following supervised models: conSultantBERTReg–ressor+RF and conSultantBERTClassifier+RF. We explore the latter since we aim to incorporate our model in a production system alongside other models and features.

4.4 Evaluation

In order to compare our different methods and baselines, we turn to standard evaluation metrics for classification problems, namely we consider ROC-AUC as our main metric, as it is insensitive to thresholding and scale invariant. In addition, we turn to macro-averaged (since our dataset is pretty balanced) precision, recall, and F1 scores for a deeper understanding of the specific behaviors of the different methods.

Finally, we perform independent student's t-tests for statistical significance testing, and set the α-level at 0.01.

5 RESULTS

See Table 1 for the results of our baselines and fine-tuned model. We first turn to our main evaluation metric ROC-AUC below, and next to the precision and recall scores in Section 5.2.

5.1 Overall performance

5.1.1 Baselines. As expected, using BERT embeddings or TF-IDF vectors in an unsupervised manner, i.e., by computing cosine similarity as a matching score, does not perform well. This confirms our observations about the challenging nature of our domain; word

overlap or semantic proximity/similarity does not seem a sufficiently strong signal for identifying which resumes match a given vacancy.

At row 3 and 4 we show the methods where a supervised classifier is trained using the input representations described above (TFIDF+RF and BERT+RF). Here, we see that with a supervised classifier, both TF-IDF-weighted vectors as feature representation and pre-trained BERT embeddings vastly improve over the unsupervised baselines, with a +31.0% improvement for BERT, and a +25.2% improvement for the TF-IDF-weighted vectors. This suggests to some extent, a mapping can be learned between the two types of documents, as we hypothesized in Section 4.2.

5.1.2 conSultantBERT. Next, we turn to our consultantBERT runs in rows 5 through 8. First, we consider models trained with the classification objective, in rows 5 and 6. We note that fine-tuning BERT brings huge gains over the results the unsupervised pre-trained embeddings (BERT+Cosine), which is in line with previous findings [14]. Even the method that relies on direct cosine similarity computations on the embeddings learned with the classification objective (conSultantBERTClassifier+Cosine), outperforms our supervised random forest baselines in rows 3 and 4 with a 7.1% and 4.2% increase in ROC-AUC respectively. Adding a random forest classifier on top of those fine-tuned embeddings (conSultantBERT Classifier+RF) even further increases performance with a +11.9% increase in ROC-AUC over our supervised baseline with pre-trained embeddings.

As our primary goal is to get high quality embeddings for the down-stream application of generating useful feature representations for recommendation, alongside of which we can train whatever model we want with additional features types such as categorical, binary, or real-valued features, we are more interested in the former unsupervised, rather than the latter approach with random forest classifier.

Finally, we turn to our fine-tuned conSultantBERT with the regression objective (conSultantBERTRegressor), to study the direct cosine similarity optimization objective. Looking at the last two rows, we find that conSultantBERT with regression objective performs similarly to conSultantBERT trained with the classification objective in supervised approach with a random forest classifier
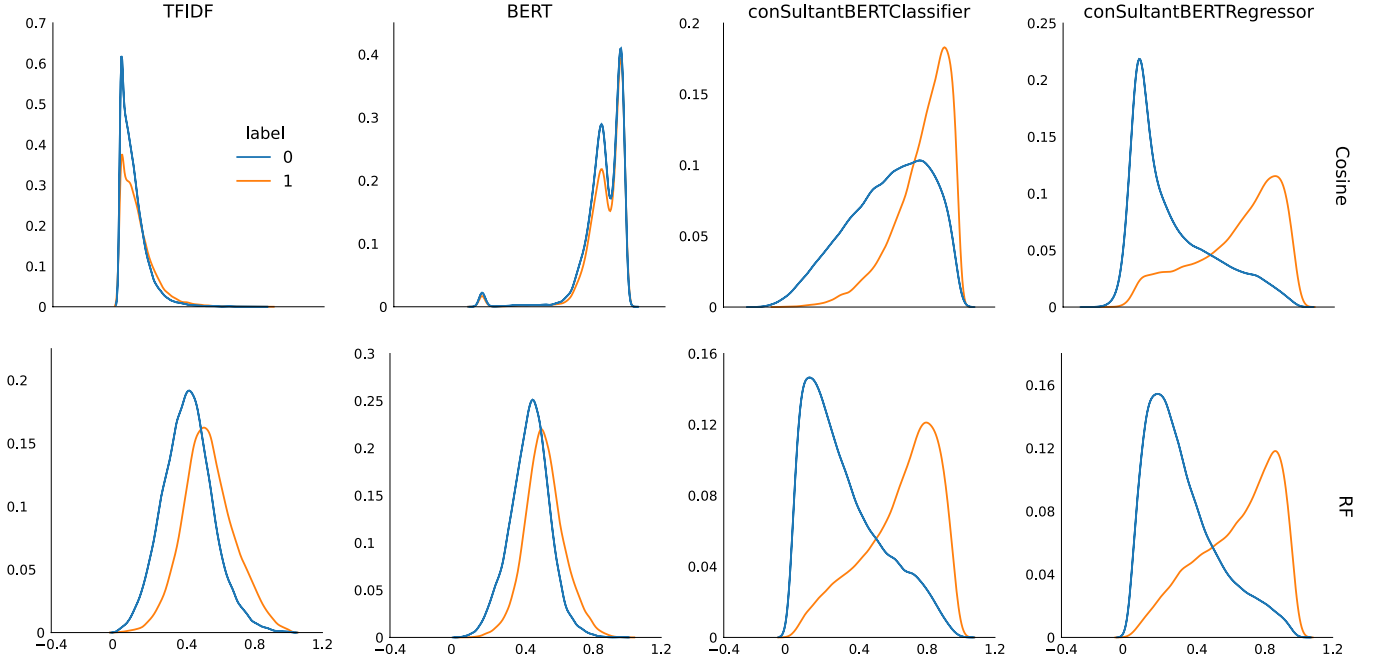
Figure 3: Density plots showing the distribution of Cosine similarity (denoted `Cosine`) scores (top row) and Random forest classifier (denoted RF) probability scores (bottom row) per label. Blue lines show the distributions for the negative class, and orange show distributions for the positive class.

on top. In fact, the small difference turns out not to be statistically significant with a p-value of 0.2.

What stands out most, is that the approach with cosine similarity outperforms all other runs, including significantly outperforming the `conSultantBERTClassifier+RF` approach. We explain this difference by the fact the architecture with the classifier objective having a learnable layer (the Softmax classifier in Figure 2). We drop this layer after the fine-tuning phase to yield our embeddings, losing information in the process. On the other hand, the architecture with the regression objective's had a non-learnable last layer, namely simply cosine similarity, which means all the necessary information has to propagate to the embedding layer.

5.2 Precision & Recall

When we zoom into the more fine-grained precision and recall scores, we observe the following: First, our unsupervised baselines in row 1 and 2 show how TFIDF-based cosine similarity yields substantially higher precision compared to cosine similarity with pre-trained BERT embeddings, keeping similar recall.

Next, these same baselines with a supervised random forest classifier on top (row 3 and 4) show that in both cases the classifiers seem to perform somewhat similarly, irrespective of whether it is trained with TFIDF-weighted feature vectors or pre-trained BERT embeddings. With only a slight increase across precision and recall (around +2.5%) for the TFIDF-based method, we conclude the feature space in itself does not provide substantially different signals for separating the classes.

Then, comparing our fine-tuned models, rows 5 and 6 shows the SBERT model fine-tuned with the classification objective yields

marginally higher precision compared to the random forest classifier trained on the SBERT's output layer. Row 7 and 8 show similar stable improvements across the board compared to the methods in rows 1–4.

Overall, we conclude that all of the supervised approaches (row 3 through 8) show roughly similar behavior with precision and recall balanced and substantial improvements over the unsupervised baselines.

6 ANALYSIS AND DISCUSSION

After analyzing the results in the previous section, in this section we take a closer look at the different methods, by studying the distributions of prediction scores across the two labels (positive and negative matches) in Section 6.1, and we zoom into another desired property of our embedding space: multilinguality (in Section 6.2)

6.1 Distributions

In Figure 3 we plot the distributions of predicted scores per run, split out for positive and negative labels. The predicted scores correspond to cosine similarities in the case of `TFIDF`, `BERT`, `conSultantBERT*`, and probabilities of the random forest classifier in the case of `TFIDF+RF`, `BERT+RF`, and `conSultantBERT*+RF`.

A few things stand out, first, it is clear that fine-tuning the BERT embeddings in our conSultantBERT models yield a clearer separation in the predictions per class. The left-most plots (TFIDF, both with Cosine similarities) show largely overlapping distributions. The second column of plots (denoted BERT) shows largely similar
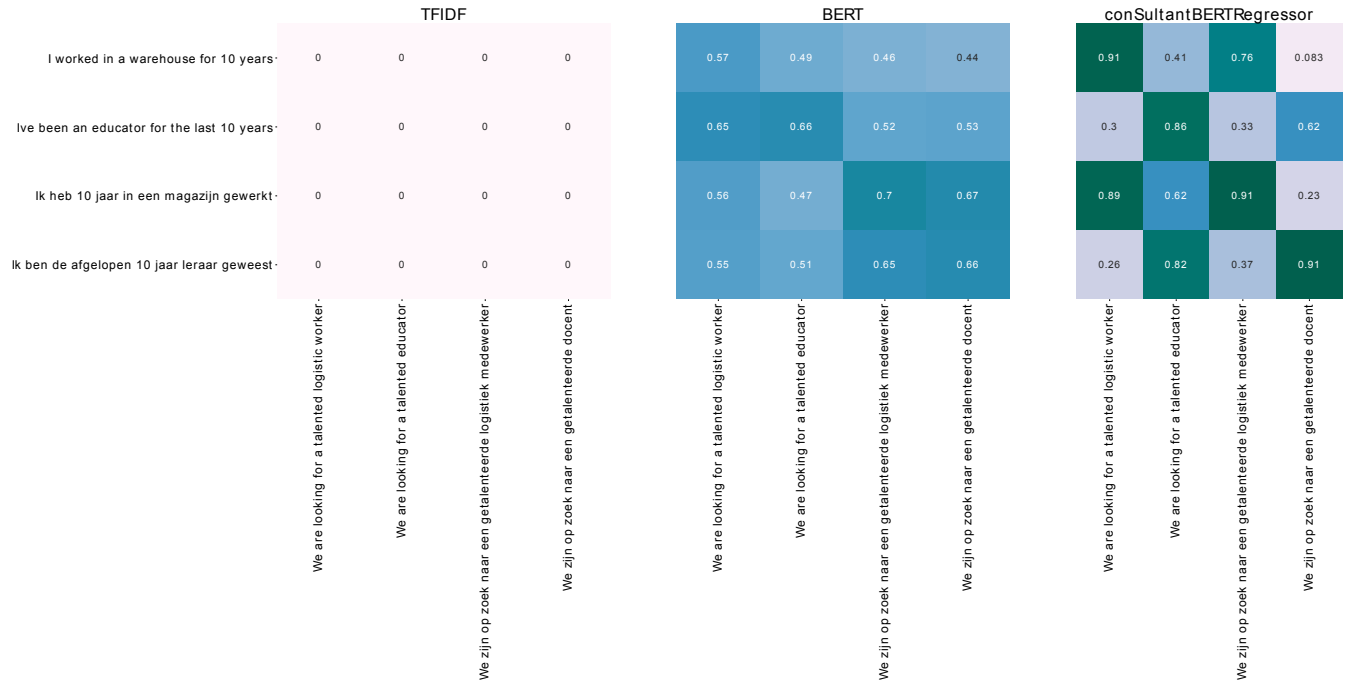
Figure 4: Heatmaps of cosine similarity between sentences from resumes and sentences from vacancies. The English sentences (first two rows, first two columns) and Dutch sentences (last two rows, last two columns) are each other's direct translations.

patterns; with as main difference the unsupervised BERT having a bias towards higher scores compared to the unsupervised TFIDF.

At the same time, while the distributions of scores for each label from the classifier with TF-IDF weighted vectors or pre-trained BERT embeddings do seem rather close, the model does succeed in separating the classes more often than not (as can be seen by the score improvements over the non-supervised baselines in Table 1).

Observing the latter, though, makes clear that unsupervised similarity scores using default embeddings or TF-IDF weights does not allow a strong signal to separate both classes, which can be witnessed by the largely overlapping score distributions.

We observed the difference in performance between our regression-optimized conSultantBERT with the classification-optimized one, in Section 5.1. We now turn to comparing the prediction distributions of both models. We note how the classification-optimized conSultantBERT (top row, third plot from the left) seems to yield less separable cosine similarity scores compared to the regression-optimized one (top right plot). Compared to the three other conSultantBERT approaches, the former stands out. The random forest classifier trained on top of the embeddings, though, effectively learns again to separate between the classes, suggesting the embeddings keep distinguishing information.

6.2 Multilinguality

As a multinational corporate that operates all across the globe, developing a model per language is not scalable in our case. Therefore, a desired property of our system is *multilinguality*; a system that will support as many languages as possible.

In some of the countries we operate, there is a high percentage of job seekers that are not native to that country. For example, many of the job descriptions in the Netherlands are in Dutch, however around 10% of the resumes are in English. Due to that fact another expected property from our solution is to be *cross-lingual* [13].

Classic text models, like TF-IDF and Word2vec, capture information within one language, but hardly connect between languages. Simply put, even if trained on multiple languages each language will have its own cluster in space. So *"logistics"* in English and *"logistiek"* in Dutch are embedded in a completely different point in space, even though the meaning is the same.

Furthermore, we know that the language of resume correlates with nationality and therefore can be a proxy discriminator. Due to the impact of these systems and the risks of unintended algorithmic bias and discrimination, HR is marked as a high risk domain in the recently published EC Artificial Intelligence Act [4]. To avoid discriminating against nationality we would like to recommend a candidate to the vacancy no matter which language the resume is written in. That is of course only if language is not a requirement for that vacancy.

In Figure 4 we see few examples for sentences that a candidate might write *"I worked ..."* and few examples for vacancy *"We are looking for ..."*. In order to demonstrate cross lingual and multilingual the same examples are written in both English and Dutch.

On the left hand side of Figure 4 (TFIDF) we can see that there is no match at all. This is due to the vocabulary gap, so by definition TFIDF can not match between *"warehouse"* and *"logistic worker"*.

To solve the vocabulary gap we introduced BERT (center of Figure 4), which we see does find similarity between candidate and

vacancy sentences. However, it also hardly separates between the positive and negative pairs. Moreover, we can see a slight clustering around languages, so Dutch sentences have comparitively higher similarity to Dutch sentences, and likewise English sentences are more similar to each other.

On the right hand side of Figure 4 (`conSultantBERTRegressor`) we observe that the vocabulary gap is bridged, cross-lingual sentences are paired properly (e.g *"I worked in a warehouse for 10 years"* and *"We zijn op zoek naar een getalenteerde logistiek medewerker"* have high score), and finally, both Dutch to Dutch and English to English sentences are properly scored, too, thus achieving our desired property of multilinguality.

7 CONCLUSION

In this work we experimented with various ways to construct embeddings of resume and vacancy texts.

We propose to fine-tune the BERT model using the Siamese SBERT framework on our large real-world dataset with high quality labels for resume-vacancy matches derived from our consultants' decisions. We show our model beat our unsupervised and supervised baselines based on TF-IDF features and pre-trained BERT embeddings. Furthermore, we show it can be applied for multilingual (e.g., English-to-English alongside Dutch-to-Dutch) and cross-lingual matching (e.g., English-to-Dutch and vice versa). Finally, we show that using a regression objective to optimize for cosine similarity yields more useful embeddings in our scenario, where we aim to apply the learned embeddings as feature representation in a broader job recommender system.

ACKNOWLEDGMENTS

REFERENCES

[1] Vedant Bhatia, Prateek Rawat, Ajit Kumar, and Rajiv Ratn Shah. 2019. End-to-End Resume Parsing and Finding Candidates for a Job Description using BERT. https://arxiv.org/abs/1910.03089

[2] Shuqing Bian, Xu Chen, Wayne Xin Zhao, Kun Zhou, Yupeng Hou, Yang Song, Tao Zhang, and Ji-Rong Wen. 2020. Learning to Match Jobs with Resumes from Sparse Interaction Data using Multi-View Co-Teaching Network. https://arxiv.org/abs/2009.13299

[3] Dhivya Chandrasekaran and Vijay Mago. 2021. Evolution of Semantic Similarity—A Survey. *ACM Comput. Surv.* 54, 2, Article 41 (Feb. 2021), 37 pages. https://doi.org/10.1145/3440755

[4] European Commission. 2021. *Proposal for a Regulation of the European Parliament and of the Council Laying Down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act) and Amending Certain Union Legislative Acts : {SEC(2021) 167 Final} - {SWD(2021) 84 Final} - {SWD(2021) 85 Final}*. European Commission. https://books.google.nl/books?id=ofxxzgEACAAJ

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805* (2018).

[6] Hebatallah A. Mohamed Hassan, Giuseppe Sansonetti, Fabio Gasparetti, A. Micarelli, and J. Beel. 2019. BERT, ELMo, USE and InferSent Sentence Encoders: The Panacea for Research-Paper Recommendation?. In *RecSys*.

[7] Po-Sen Huang, Xiaodong He, Jianfeng Gao, li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. 2333–2338. https://doi.org/10.1145/2505515.2505665

[8] Cataldo Musto, Giovanni Semeraro, Marco de Gemmis, and Pasquale Lops. 2016. Learning Word Embeddings from Wikipedia for Content-Based Recommender Systems. In *Advances in Information Retrieval*, Nicola Ferro, Fabio Crestani, Marie-Francine Moens, Josiane Mothe, Fabrizio Silvestri, Giorgio Maria Di Nunzio,

Claudia Hauff, and Gianmaria Silvello (Eds.). Springer International Publishing, Cham, 729–734.

[9] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[10] Chuan Qin, Hengshu Zhu, Tong Xu, Chen Zhu, Liang Jiang, Enhong Chen, and Hui Xiong. 2018. Enhancing Person-Job Fit for Talent Recruitment: An Ability-aware Neural Network Approach. *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval* (2018).

[11] Rohan Ramanath, Hakan Inan, Gungor Polatkan, Bo Hu, Q. Guo, C. Ozcaglar, Xianren Wu, K. Kenthapadi, and S. C. Geyik. 2018. Towards Deep and Representation Learning for Talent Search at LinkedIn. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (2018).

[12] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 3982–3992. https://doi.org/10.18653/v1/D19-1410

[13] Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2019. A Survey of Cross-Lingual Word Embedding Models. *J. Artif. Int. Res.* 65, 1 (May 2019), 569–630. https://doi.org/10.1613/jair.1.11640

[14] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to Fine-Tune BERT for Text Classification?. In *Chinese Computational Linguistics*, Maosong Sun, Xuanjing Huang, Heng Ji, Zhiyuan Liu, and Yang Liu (Eds.). Springer International Publishing, Cham, 194–206.

[15] Jeff Weiner. 2012. The Future of LinkedIn and the Economic Graph. (2012).

[16] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. HuggingFace's Transformers: State-of-the-art Natural Language Processing. arXiv:1910.03771 [cs.CL]

[17] Jing Zhao, Jingya Wang, Madhav Sigdel, Bopeng Zhang, Phuong Hoang, Mengshu Liu, and Mohammed Korayem. 2021. Embedding-based Recommender System for Job to Candidate Matching on Scale. https://arxiv.org/abs/2107.00221

[18] Chen Zhu, Hengshu Zhu, Hui Xiong, Chao Ma, Fang Xie, Pengliang Ding, and Pan Li. 2018. Person-Job Fit: Adapting the Right Talent for the Right Job with Joint Representation Learning. https://arxiv.org/abs/1810.04040