What Company Does My News Article Refer to? Tackling Multiclass Problems With Topic Modeling

Max Lübbering[1][0000−0001−6291−9459], Julian Kunkel[2][0000−0002−6915−1179], and Patricio Farrell[3][0000−0001−9969−6615]

[1] Fraunhofer IAIS, Sankt Augustin, Germany
max.luebbering@iais.fraunhofer.de
[2] University of Reading, Reading, United Kingdom
[3] Weierstrass Institute (WIAS), Berlin, Germany

Abstract. While it is technically trivial to search for the company name to predict the company a new article refers to, it often leads to incorrect results. In this article, we compare the two approaches *bag-of-words with k-nearest neighbors* and *Latent Dirichlet Allocation with k-nearest neighbor* by assessing their applicability for predicting the S&P 500 company which is mentioned in a business news article or press release. Both approaches are evaluated on a corpus of 62k documents containing 84% news articles and 16% press releases. While the *bag-of-words* approach yields accurate predictions, it is highly inefficient due to its gigantic feature space. The *Latent Dirichlet Allocation* approach, on the other hand, manages to achieve roughly the same prediction accuracy (0.58 instead of 0.62) but reduces the feature space by a factor of seven.

Keywords: Text Classification · Latent Dirichlet Allocation · Kullback-Leibler Divergence · Company Prediction · News Articles.

1 Introduction

Labeling a news article with the company that it deals with, is not an easy classification task. In this paper, we develop a model that a) classifies news articles with hundreds of target labels (i.e., companies), b) does not require retraining for every additional target label and c) supports anonymized news articles (i.e., no company or stock symbol mentions).

To solve this classification problem, we evaluate two alternative approaches, namely *bag-of-words with k-nearest neighbors* and *Latent Dirichlet Allocation with k-nearest neighbor*, on a large data set of news. The first approach spans a huge feature space as every feature of the bag-of-words model is directly passed to the k-nearest neighbor classifier, leading to an overall minimally better performance than the second approach, however, being highly inefficient. The second

approach transforms the features of the bag-of-words to a new set of features (topics in the training corpus), which – as shown in the evaluation – is smaller by a factor of seven. The feature transformation is based on Blei's Latent Dirichlet Allocation (LDA) [3].

While state-of-the-art approaches seem more applicable than LDA, they do not fulfill all of the three model requirements stated above. Named entity recognition with subsequent company name matching would fail for anonymized texts. LSTMs (especially attention based LSTMs) would work for longer texts, but require retraining. Even though models based on embedding representation could fulfill all three requirements, they have poor clustering abilities. Pre-trained word embedding models like Word2Vec [7] and character level word embedding models like Flair [1], can encode syntactic and semantic word-level properties. However their clustering and classification capabilities (especially on document level) are rather limited [4]. The two models are trained on word and character substitutability respectively, which does not provide distinct topical clusters in its vanilla form [5]. Thus requiring further downstream classification layers, that need to be retrained with every additional target label.

This article is structured in five sections: In Section 2, we first present text classification methods known in the literature and subsequently detail our strategies to combine some of them. Section 3 deals with the implementation of the two proposed approaches from the previous section. The two approaches are further evaluated on multiple data sets in Section 4. These results are discussed and put into perspective in Section 5

2 Method

Before presenting our two specific approaches for predicting the company a news article refers to in Section 2.2, we present the work other researchers have performed on text classification.

2.1 Related Work

Topic based text classification has been covered in different research areas already [11,8].

A possible way to preprocess documents is to represent them by a topic model as done by the authors of [9,10]. For a finite set of topics, each document's topic representation is a categorical distribution over the topics.

In contrast to [10], whose authors picked a specific corpus of clinical reports, the authors in [9] chose universal datasets (e.g., a subset of the Wikipedia article corpus) to train a topic model. The topic model then predicted the topics of a labeled data set, which were then used as input to train multiple classifiers e.g., naïve Bayes or support vector machines (SVMs). The model they trained were able to classify short and sparse texts.

The application of domain specific corpora to text classification has been shown by Sarioglu et al. [10]. They built a topic model from clinical reports in

order to represent them in a more compact feature space than one built from a bag-of-words model. The representations were then used to classify CT[4] imaging reports using SVMs. Their results showed, that the topic model approach was competitive with a bag-of-words approach, while reducing the number of features significantly.

2.2 Design

We train our two approaches on the company descriptions corpus D and evaluate the models on the news article corpus A, as shown in Figure 1. Each company description $\mathbf{d}_i$ and news article $\mathbf{a}_i$ contains the respective document in its human-readable text representation. In both corpora each document is labeled by the respective company name.

Fig. 1: Document corpora

Our two approaches classify news articles and press releases[5] by predicting their similarity to a set of company descriptions, which are labeled by the corresponding company name. In order to reduce complexity, our approaches will be trained to predict only one company per news article and the set of companies is limited to the companies (denoted by C) listed in the S&P 500 index.

Next, we explain our two classification approaches. The ***bag-of-words with k-nearest neighbor (BOW KNN)*** approach is the trivial one, that we want to beat by the second approach with respect to prediction accuracy and model complexity. The second approach, namely ***Latent Dirichlet Allocation with k-nearest neighbor (LDA KNN)***, is more sophisticated and can be seen as an extension of the first one. The advantage of this extension is its capability to reduce the set of features tremendously, as we will see later in this article.

As the k-nearest neighbor models in both approaches will be trained on categorical distributions which are defined on a simplex, Euclidean distance is not applicable as a metric. This is why we use the Kullback Leibler divergence (KL) to determine the similarity between two categorical distributions p and q according to

$$KL(p||q) = -\sum_{x \in X} p(x) \ln\left(\frac{q(x)}{p(x)}\right) \quad \in [0, \infty), \tag{1}$$

where X is the set of all categories (here, the set of all words in vocabulary $\mathbf{v}$). Note, that in contrast to the Euclidean norm, the Kullback Leibler divergence

[4] computed tomography

[5] In the following, for simplicity we refer to press releases as news articles as well.

is not a metric (in the mathematical sense) and is used as a similarity measure, where a value of zero means maximum similarity.

2.3 Bag-of-words with k-nearest neighbor

Bag-of-words with k-nearest neighbor (BOW KNN) first builds a vocabulary (set of all distinct words in a corpus) from the company description corpus D and counts the number of occurrences of each word for each document, which can be represented by a matrix

$$\tilde{D}_{M,|\mathbf{v}|} = \begin{pmatrix} \tilde{d}_{1,1} & \tilde{d}_{1,2} & \cdots & \tilde{d}_{1,|\mathbf{v}|} \\ \tilde{d}_{2,1} & \tilde{d}_{2,2} & \cdots & \tilde{d}_{2,|\mathbf{v}|} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{d}_{M,1} & \tilde{d}_{M,2} & \cdots & \tilde{d}_{M,|\mathbf{v}|} \end{pmatrix} \in \mathbb{R}^{M \times |\mathbf{v}|}, \tag{2}$$

where M is the number of company descriptions in the corpus, $|\mathbf{v}|$ is the length of the vocabulary $\mathbf{v}$ and an element $\tilde{d}_{i,j}$ represents how often word j appears in company description i. Note, that usually $|\mathbf{v}| \gg M$. This absolute word frequencies representation of the company descriptions is called bag-of-words representation [6]. In the following, we always normalize the bag-of-words representation matrix of a corpus in a row-wise fashion. For the bag-of-word representation $\tilde{D}$ of the company descriptions D this means, that every element in the matrix is divided by its document's length: $\tilde{d}_{i,j} \to \frac{\tilde{d}_{i,j}}{\sum_{j=1}^{|\mathbf{v}|} \tilde{d}_{i,j}}$. Since the normalization ensures $\sum_{j=1}^{|\mathbf{v}|} \tilde{d}_{i,j} = 1$ and $0 \leq \tilde{d}_{i,j} \leq 1$, every document is represented by a categorical distribution over the vocabulary.

In the following, we will assert every bag-of-words model to be present in its normalized form.

As shown in Figure 2, the idea of *BOW KNN* is to build a bag-of-words model from the company description corpus D (i.e, determine $\mathbf{v}$) and represent the corpus in terms of the bag-of-words model, leading to matrix $\tilde{D}$. These representations are then used to train the k-nearest neighbor classifier (KNN classifier).

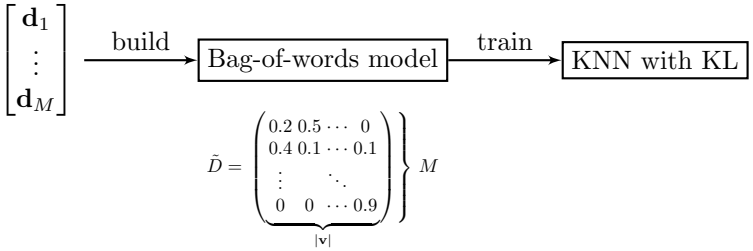


Fig. 2: Bag-of words k-nearest neighbor model training

For news article prediction, the news article $\mathbf{a}_i$ is represented in terms of the same bag-of-words model and then passed to the k-nearest neighbor classifier, which determines the k best matching company descriptions for the given article $\mathbf{a}_i$. This process is shown in Figure 3.

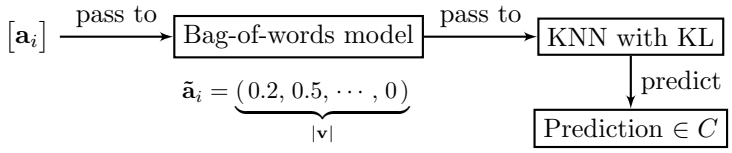


Fig. 3: Bag-of words k-nearest neighbor model prediction

The *BOW KNN* approach has certain disadvantages:

- Bag-of-words models often have a very large feature space (dimensionality = $|\mathbf{v}|$), which makes similarity calculations between documents expensive and time consuming.
- Unimportant features add a lot of noise to the similarity measurement, thus leading to models that do not generalize well in practice.

2.4 Latent Dirichlet Allocation with k-nearest neighbor

While feature selection strategies like mutual information and χ^2 test statistic have been successfully used for years to reduce the dimensionality by identifying unimportant features, Latent Dirichlet Allocation (LDA), which is part of our second approach, generates a complete new set of features, namely the topics of a corpus.

Using the *Latent Dirichlet Allocation with k-nearest neighbor (LDA KNN)* approach we try to solve the previously listed shortcomings. We will analyze how *LDA KNN* can reduce the huge feature space of dimensionality $|\mathbf{v}|$ spanned by the bag-of-words model, while incurring only small losses in accuracy. This approach takes the normalized bag-of-words representation $\tilde{\mathbf{d}}_i$ of the company descriptions and trains a topic model using LDA.

Each of these topics $\boldsymbol{\beta}_i$ of the topic model is a categorical distribution over the vocabulary $\mathbf{v}$, where the importance of a word for a topic is expressed by the amount of probability assigned to the word. The modeling task in LDA is to determine the categorical distribution for each topic $\boldsymbol{\beta}_i$ and the parameterization $\boldsymbol{\alpha}$ of the Dirichlet distribution (see [2]), such that the log likelihood $l(\boldsymbol{\alpha}, \beta)$ of the company description corpus is maximized:

$$\arg\max_{\boldsymbol{\alpha},\beta} l(\boldsymbol{\alpha}, \beta) = \arg\max_{\boldsymbol{\alpha},\beta} \sum_{i=1}^{M} \log(p(\tilde{\mathbf{d}}_i|\boldsymbol{\alpha}, \beta)). \tag{3}$$

Note, that the parameter β is the set of all topics and $p(\tilde{\mathbf{d}}_i|\boldsymbol{\alpha}, \beta)$ is the probability of company description $\tilde{\mathbf{d}}_i$ given the model parameters $\boldsymbol{\alpha}$ and β.

We set the number of topics (hyperparameter) in the LDA model to match the number of all companies in C. **Since the company descriptions belonging to the same company are most similar to each other in terms of their words, we thereby create an artificial bias towards representing every company by exactly one topic**, albeit articles of different companies may be similar, e.g., articles from the same industry like a car manufacturer.

Having trained the LDA model on the company description corpus, i.e. estimated β and $\boldsymbol{\alpha}$, we predict the topic distribution for every company description by the LDA model, as shown in Figure 4. This yields a normalized topic prediction matrix $\bar{D}$ of dimensions $M \times |topics|$, with $0 \leq \bar{d}_{i,j} \leq 1$ and $\sum_{j=0}^{|topics|} \bar{D}_{i,j} = 1$ for all $i \in \{0, 1, \ldots, |topics|\}$. The matrix entry at position (i, j) denotes how much the j^{th} topic is represented in the i^{th} company description in relation to all the other topics.

The k-nearest neighbor classifier is trained on $\bar{D}$. As $|companies| \ll |\mathbf{v}|$, the feature space of the KNN classifier is reduced significantly compared to *BOW KNN*.

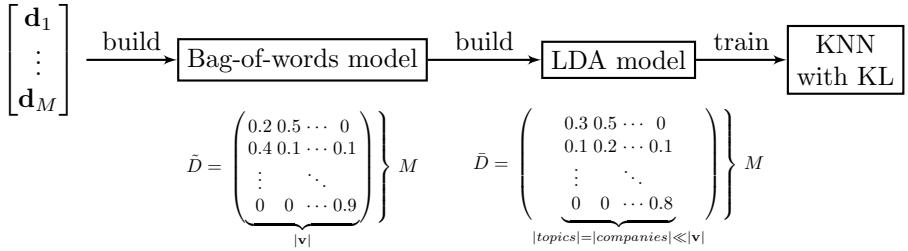


Fig. 4: Latent Dirichlet Allocation k-nearest neighbor model training

As shown in Figure 5, in order to predict its company, the news article is first represented in terms of the bag-of-words model, then by the LDA topic model and finally passed to the KNN classifier. The KNN classifier picks the k most similar company descriptions by calculating the Kullback-Leibler divergence between the topic distribution $\bar{\mathbf{a}}_i$ of the news article and the topic distributions of all company descriptions. We then predict the company, whose company descriptions were selected most frequently among the k closest ones. Note, that both approaches also allow to quantify a prediction's certainty. When all of the k neighbors are the same, the model is 100%, whereas when two companies are predicted four and six times out of $k = 10$, then the model is only 40% certain.

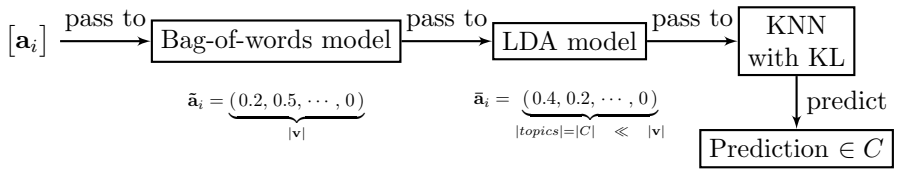


Fig. 5: Latent Dirichlet Allocation k-nearest neighbor model prediction

3 Implementation

In this section, we explain the implementation of the methodology proposed in the previous section by means of the general machine learning process: data acquisition, preprocessing, and modeling.

3.1 Data Acquisition

To train our models, we need multiple company descriptions per company, which is why we implemented a company description crawler, that retrieves the company descriptions for every company in the S&P 500 index from a total of eight relevant financial information providers like Yahoo! Finance or Google Finance. In order to evaluate the models' performance in the field, we crawled news articles from different outlets. These news articles were unlabeled, i.e., did not indicate the company name(s), and therefore, not applicable for evaluation. To label them, our first approach was to search Google Finance and Google News by company name or company stock symbol. For each crawled news article URL that matched one in our news article data set, we were able to label the news article by the company name. This approach yielded only 666 labeled articles (see Section 4.2), therefore, we labeled articles using a keyword search in a second approach (trivial labeler). For each company, the number of its occurrence in a news article is being counted. The article was labeled by the company contained most often in the article. Manual inspection revealed many false negatives, which is why we put further constraints on an article to be successfully labeled: The top company has to appear at least three times in an article, the total number of companies mentioned in the article has to be less than four and the top company has to be mentioned at least two more times than the second most mentioned company. Applying these constraints, we manually determined a proportion of 99% correctly labeled news articles in a randomly generated subset of 400 news articles. All documents were retrieved in plain HTML, which is why we applied a two layered preprocessing: 1) Parse HTML tree to extract the article text. 2) The text is tokenized, POS tagged, lemmatized and stop-word cleaned.

3.2 Modeling

Having implemented the pipelines of both approaches using Python's scikit-learn library[6], we needed to hyperparameterize both pipelines, before training the models. Since we achieved for different hyperparameterizations huge differences in the accuracy of our validation set, we decided to train multiple models on a variety on parameter combinations in a grid search fashion. In Table 1, we list all tested values for each parameter.

Bag-of-words with KNN

- Bag-of-words modeling
  - `max_df`: [0.025, 0.05, 0.1, 0.2, 0.4]
  - `min_df`: [1, 3, 5, 7, 10]
- KNN modeling
  - `n_neighbors`: [3, 8, 15, 20]
  - `weights`: [distance, uniform]
  - `metric`: [KL, KL']

LDA with KNN

- Bag-of-words modeling
  - `max_df`: [0.025, 0.05, 0.1, 0.2, 0.4]
  - `min_df`: [1, 3, 5, 7, 10]
- LDA modeling
  - `n_components`: [480, 1000, 2000, 3000, 5000, 10000]
  - `learning_method`: batch
- KNN modeling
  - `n_neighbors`: [3, 8, 15, 20]
  - `weights`: [distance, uniform]
  - `metric`: [KL, KL']

Table 1: Hyperparameterization combinations passed to the grid search algorithm for the two approaches *BOW KNN* and *LDA KNN*. The hyperparameter names are consistent with the definitions in the scikit-learn library. KL is the Kullback Leibler divergence and KL' is the Kullback Leibler divergence, where the arguments are swapped.

Since words that are existing in nearly all or none of the articles are irrelevant for the analysis, we set all considered tokens to be in a pre-defined document frequency range, when building the bag-of-words model. The hyperparameter `max_df` limits vocabulary to the words, that have a lower document frequency, than the given maximum. Accordingly, `min_df` defines the lower bound. Note, that if we pass absolute values to any of the two hyperparameters, then they are not considered as document frequencies, but as absolute appearances in the documents.

KNN is parameterized such that for classification the k most similar neighbors (`n_neighbors`) are being picked based on Kullback Leibler divergence

[6] `http://scikit-learn.org/stable/index.html`

(`metric`)[7]. Classification is done by either weighting each of the closest neighbors by their distance to the sample or by weighting them uniformly (`weights`).

For LDA modeling, we have to pass the number of topics (`n_components`) and picked *batch* as the learning method to go with.

4 Results

4.1 Accuracy Measure

We evaluate the models' performance by the accuracy measure

$$a(model) = \frac{\sum_{l=0}^{|L|} \frac{TP_l}{|X_l|}}{|L|}, \tag{4}$$

where L is the set of all classes present in the test data set X. The variable TP_l is the number of instances in X having true class l, that have been also predicted as l (true positives). The subset X_l of X contains the instances, that belong to class l. This measure does not take the true negatives (TN) into account, as these instances are almost always correctly predicted due to the large size of L and thereby would lead to misleading accuracies.

4.2 Datasets

After having trained the models on the company descriptions data set, their performance is subsequently evaluated on the company descriptions data set and three additional data sets:

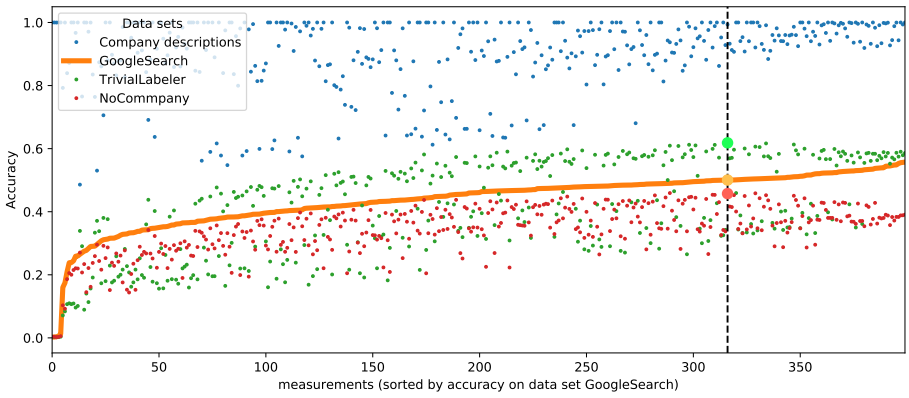
- **GoogleSearch**: **News article set labeled by Google search**
  Each news article crawled by the the news article crawler (in total 2.2 million), was labeled by matching it against Google results when searching for the company names. The resulting data set contains 666 news articles, while 55% of these news articles are press releases. The data set covers 275 companies of the S&P 500 index.
- **TrivialLabeled**: **News article set labeled by trivial labeler**
  Since the first data set turned out to be rather small, we decided to label the 2.2 million news articles using keyword search for company names, as explained in Section 3.1. The resulting data set contains 62,000 labeled news articles of which 16% are press releases. It covers 458 companies of the S&P 500 index while being highly imbalanced. To compensate for this, we limited the number of news articles per company to 50. The final data set covered 13,000 articles.
- **NoCompany**: **News article set without company names**
  For this data set, we used the *TrivialLabeled* data set and deleted all company names and stock symbols. The prediction accuracy for this data set, gives insights whether the models have learned information about a company besides relying solely on, e.g., company name and stock symbol.

[7] Note, that the library scikit-learn called the parameter "metric", even though it is a similarity measure and not a metric in the mathematical sense.

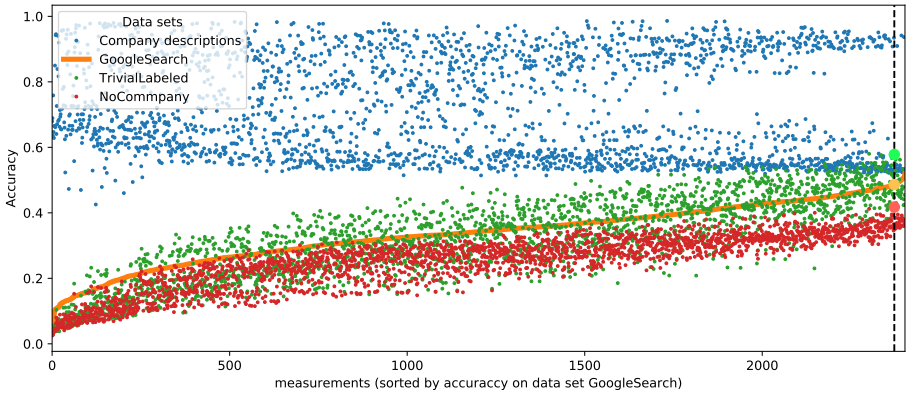
4.3 Evaluation

Having performed grid search, the results have been sorted by the accuracy for the data set *GoogleSearch*, as shown in Figure 6. The dotted vertical line with the highlighted points shows the parameter combinations which provided the highest accuracy over the three data sets.

There is a strong correlation between all three test sets, whereas the *company descriptions* data set especially for the *LDA KNN* shows no correlation at all. The data set *TrivialLabeled*, shows an overall better accuracy than the data set *NoCompany*, while the accuracy rate is still high for well chosen parameters. Therefore, the models exploit the knowledge about company names and stock symbol to some degree. The exact accuracies for the two best classifiers are



(a) BOW KNN grid search results



(b) LDA KNN grid search results

Fig. 6: Grid search results

given in Table 2. For all three data sets the best *BOW KNN* classifier exceeds the best *LDA KNN* classifier by few percent points in terms of accuracy. The best *BOW KNN* classifier's parameterization is `max_df`=0.1, `min_df`=1 (bag-of-words modeling) and `n_neighbors`=20, `weights`=distance, `metric`=KL' (KNN modeling). For the best *LDA KNN* classifier we got the following parameterization: `max_df`=0.2, `min_df`=1 (bag-of-words modeling) and `n_components`=3000 (LDA modeling) and `n_neighbors`=20, `weights`=distance, `metric`=KL' (KNN modeling). The accuracies of the *BOW KNN* model trained on a vocabulary of size 3,102 is shown in the parentheses for each data set in Table 2.

Data set	***LDA KNN***	***BOW KNN***
Google search	0.49	0.50 (0.40)
TrivialLabeled	0.58	0.62 (0.43)
NoCompany	0.42	0.46 (0.31)

Table 2: Accuracy on different data sets according to Equation (4) of best *LDA KNN* model trained on 3,000 topics and best *BOW KNN* trained on vocabulary of size 21,700 and a second *BOW KNN* model trained on vocabulary of size 3,102 (see accuracy values in parentheses)

.

5 Discussion

We have shown, that both approaches work in the field and exceed random company guessing by at least 263 times on the data set *TrivialLabeled* after a proper grid search parameter optimization. On all data sets, the *BOW KNN* outperforms *LDA KNN* only by about three percent points in terms of accuracy (Equation (4)).

Additionally, the LDA part in *LDA KNN* reduces the feature space from 21,700 to 3,000 features which is a reduction by 86%. Training *BOW KNN* on a similar feature space as the best *LDA KNN* model (i.e., $\sim 3,000$ features), yields poor results in terms of accuracy, making *LDA KNN* clearly superior for smaller feature spaces.

As stated in Section 2.2, the intention was to create a bias (due to the company descriptions) such that every company would be represented by exactly one topic. Our grid search results, showed that this bias was not strong enough and lead to an optimal number of topics, that was about six times higher than expected.

Furthermore we have shown, that both algorithms can learn information about a company other than its plain company name or its stock symbol. This is a huge advantage over, algorithms that perform a simple keywords search for the company name. This insight is important when, e.g., texts have been deliberately anonymized or only the products not its producer are named in an

article. In this case a simple keyword search by company or a NER approach would not yield any results, while *BOW KNN* and *LDA KNN* still provide valuable results.

References

1. Akbik, A., Blythe, D., Vollgraf, R.: Contextual string embeddings for sequence labeling. In: COLING 2018, 27th International Conference on Computational Linguistics. pp. 1638–1649 (2018)
2. Blei, D.M.: Probabilistic topic models. Commun. ACM **55**, 77–84 (2012), http://doi.acm.org/10.1145/2133806.2133826
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research **3**, 993–1022 (2003)
4. Diaz, F., Mitra, B., Craswell, N.: Query expansion with locally-trained word embeddings. arXiv preprint arXiv:1605.07891 (2016)
5. Liu, Q., Huang, H., Gao, Y., Wei, X., Tian, Y., Liu, L.: Task-oriented word embedding for text classification. In: Proceedings of the 27th International Conference on Computational Linguistics. pp. 2023–2032 (2018)
6. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval, p. 117. Cambridge University Press (2008)
7. 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)
8. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: Sentiment classification using machine learning techniques. In: Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10. pp. 79–86. Association for Computational Linguistics (2002), https://doi.org/10.3115/1118693.1118704
9. Phan, X.H., Nguyen, L.M., Horiguchi, S.: Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In: Proceedings of the 17th International Conference on World Wide Web. pp. 91–100. ACM (2008), http://doi.acm.org/10.1145/1367497.1367510
10. Sarioglu, E., Yadav, K., Choi, H.A.: Topic modeling based classification of clinical reports. In: 51st Annual Meeting of the Association for Computational Linguistics Proceedings of the Student Research Workshop. pp. 67–73 (2013)
11. Sriram, B., Fuhry, D., Demir, E., Ferhatosmanoglu, H., Demirbas, M.: Short text classification in twitter to improve information filtering. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 841–842. ACM (2010), http://doi.acm.org/10.1145/1835449.1835643