Surveying public opinion using label prediction on social media data

Marija Stanojevic Jumanah Alshehri Zoran Obradovic
Center for Data Analytics and Biomedical Informatics
Temple University
Philadelphia, PA, USA
{marija.stanojevic, jumanah.alshehri, zoran.obradovic}@temple.edu

***Abstract*—In this study, a procedure is proposed for surveying public opinion from big social media domain-specific textual data to minimize the difficulties associated with modeling public behavior. Strategies for labeling posts relevant to a topic are discussed. A two-part framework is proposed in which semi-automatic labeling is applied to a small subset of posts, referred to as the "seed" in further text. This seed is used as bases for semi-supervised labeling of the rest of the data. The hypothesis is that the proposed method will achieve better labeling performance than existing classification models when applied to small amounts of labeled data. The seed is labeled using posts of users with a known and consistent view on the topic. A semi-supervised multi-class prediction model labels the remaining data iteratively. In each iteration, it adds context-label pairs to the training set if softmax-based label probabilities are above the threshold. The proposed method is characterized on four datasets by comparison to the three popular text modeling algorithms (n-grams + tfidf, fastText, VDCNN) for different sizes of labeled seeds (5,000 and 50,000 posts) and for several label-prediction significance thresholds. Our proposed semi-supervised method outperformed alternative algorithms by capturing additional contexts from the unlabeled data. The accuracy of the algorithm was increasing by (3-10%) when using a larger fraction of data as the seed. For the smaller seed, lower label probability threshold was clearly a better choice, while for larger seeds no predominant threshold was observed. The proposed framework, using fastText library for efficient text classification and representation learning, achieved the best results for a smaller seed, while VDCNN wrapped in the proposed framework achieved the best results for the bigger seed. The performance was negatively influenced by the number of classes. Finally, the model was applied to characterize a biased dataset of opinions related to gun control/rights advocacy. The proposed semi-automatic seed labeling is used to label 8,448 twitter posts of 171 advocates for guns control/rights. On this application, our approach performed better than existing models and it achieves 96.5% accuracy and 0.68 F1 score[1].**

***Index Terms*—semi-supervised, label prediction, social media**

I. Introduction

Understanding public opinion helps in making more informed decisions, policies, and products. Thanks to social networks, forums, blogs, product reviews and news comments, it is easy to obtain a large amount of text containing opinions about specific topics. Social networks became primary platforms for the public debates where anyone can share their opinion [1–5].

[1]First two authors have contributed equally to this work.

However, surveying public opinion in a traditional way is a costly and time-consuming process which can require contacting many people. Using online data is an appealing alternative, but such text is typically noisy and biased. These limitations can be minimized through careful data collection to ensure that relevant posts are included, and non-relevant posts are discarded [6].

When using social media, news comments, forum posts, and product reviews to model an opinion, researchers face a significant constraint because such texts are very short. Additionally, they often contain pictures, links, and emoticons that can strongly influence meaning or change the tone of the text. To overcome those challenges and achieve increased accuracy, recent papers proposed text classification models that require hundreds of thousands or millions of labeled documents [7]. To model public opinion in this way, a massive amount of data has to be collected together with their labels that represent different views on the same topic.

It is hard and expensive to label huge amount of posts. Most of academic and industry organizations don't have the resources to support labeling of so much data. Even if resources are available, it may take years to label enough data for certain problems. Moreover, in cases when the task of labeling is too complex or the example is very short or has layered meanings, experts on the topic are needed. If labeling can be automated, more focus can be given to modeling human behaviour and opinions.

One way of automating the labelling process is to cluster documents [8–11]. However, documents could be clustered on some other characteristics and not the view on a particular topic of interest and therefore, there is no guarantee that clusters would contain different opinions.

In this study, we address this problem by a Semi-supervised Label Prediction (SLP), an easy and efficient semi-supervised approach that predicts labels of vast numbers of documents based on the small seed (documents that are labeled). Additionally, we describe how to quickly label a seed. We provide evidence that our model can predict labels more accurately than existing classification methods when keeping time complexity the same.

Contributions of this study include the following:

1) A semi-supervised label prediction (SLP) algorithm is

proposed to label big textual data using a small labeled seed.
2) A semi-automatic labeling framework is proposed to label the seed in order to decrease labeling effort for an order of magnitude.
3) SLP is evaluated on multiple datasets using a range of seed size and label prediction significance thresholds. It outperformed all existing solutions on all of the datasets considered.
4) The SLP framework is successfully applied to characterize public opinion on gun advocacy using possibly biased twitter posts of 200 gun control/rights advocates.

II. Related work

Related work for the proposed approach can be separated into three main components: language modeling that is used to convert the documents to vectors, training using labeled document vectors (text classification) and semi-supervised label prediction.

In a character-level convolution model [12] it is shown that modeling language as a bag of character ngrams, calculating tfidf of those ngrams and doing logistic regression achieves the best results on classification tasks for smaller datasets (below 600K examples). Therefore, we use this model as a part of our framework and as a baseline, referring to it as **tfidf + ngrams**.

Word2vec [13] embeds words in a self-supervised way by predicting context around the word (skip-gram) or word from the context (CBOW). It is able to better capture syntax and semantic meaning of the words than tfidf scores of words. Global vectors for word representation (**GloVe**) [14] captures the meaningful linear substructures by matrix decomposition of a word-word concurrence matrix and performs even better in syntax and semantic meaning understanding. However, both of those models require additional steps to create a document vector such as word vector averaging which can lower their power and which heavily depends on the text content, length and cleaning process. **Doc2vec** algorithm is able to learn representations of larger blocks of text (documents) together with learning representations of words in it [15]. Nonetheless, our experiments suggest that doc2vec doesn't perform well on short and domain-specific texts.

To improve task accuracy, many algorithms integrally model and classify the texts (end-to-end approach). Many solutions use a long short-term memory approach (**LSTM**) in unidirectional or bidirectional way on characters or words [16]. Some are trying to improve accuracy with multi-task learning [17]. Due to the sequential component of the text, this is a natural choice where bidirectional LSTM (bi-LSTM) approaches are performing well since both previous and future context is important. However, with longer input sequences, the number of weights to learn becomes big and therefore it requires more data. Additionally, it requires more time to train and gradients issues start appearing and decreasing model performance.

To avoid those issues, research turned to the convolutional neural networks (CNN). **Character-level convolutional network** achieves very good accuracy for datasets with more than 600K examples [12]. Even better results were obtained using a combination of convolutional layers and bi-LSTM to learn from context [18]. Convolutional layer learns higher level features which produces a short sequence that is then used as input of bi-LSTM. Since computer vision benefited from using very deep CNN (**VDCNN**) a similar method is applied to text. [7]. VDCNN consists of convolutional blocks. Each block contains two convolutional layers followed by a batch normalization and ReLU. Block output can be connected with it's input as in resNet [19]. This architecture achieved better results than all previous methods for datasets bigger than 500K examples. Even though all of these methods can be part of our framework, we use VDCNN as it has the best performance. We also use it as a baseline.

Recent approaches combine deep models with attention layers. For example, hierarchical attention network (**HAN**) [20] uses bidirectional layers with GRU units, and it consists of five components. Sequence encoder maps word/sentence into a vector and then gets annotation of words/sentences by summarizing information from the forward and backward hidden state. The Word/sentence level attention layer extracts the important words/sentences and aggregates the representation into sentence/document vectors that are classified in the last step. **fastText** classifies documents using weighted word embedding and it achieves similar performance as the previous models. Thanks to hashing, this model is significantly faster than others. We use it as a component in our framework and as a baseline. **StarSpace** model is an extension of fastText that learns entities embedding with discrete feature representation using additional entities beside text. This approach was shown to improve classification results a bit and it can be used as part of our framework if additional information is known besides the text [21].

Semi-supervised label prediction. Traditional semi-supervised label prediction models, such as co-training [22], naive bias classifier and expectation-maximization [23] worked with much longer texts and required additional information about it or appropriate mathematical representation to optimize. Newer methods surveyed in [24] are mainly based on matrix factorization. None of these models can be combined with deep learning models. Some studies [25, 26] label words as positive/negative. Based on the words used in the text they assign a label to the document. In social media posts, however, words are used with different meanings, or they are written with typographic errors, or with a cynical tone, and therefore it would be difficult to make appropriate word labeling.

III. Proposed Methods

A. Seed Labeling

Labeling a small amount of posts (seeds) that can be used for initial model training can still require significant effort and money. Since most social media platforms don't allow data sharing, every entity has to collect and label data separately.

In [27] it is shown that nearly 90% of data, posted by users who are known to be active in discussing the topic, were relevant to the topic. Therefore, the easiest way to label posts for seed is to find profiles of users and organizations which advocate a position equivalent to each of the labels. For

more significant political and social discussions, lists of those organizations can be found online. Additionally, most social platforms have recommendation and search systems that show similar profiles/organizations or can list organizations relevant for a phrase.

It is important to extract only profiles that have one opinion on the topic of interest so that we can label all the posts of that user with the same class. Profiles of media and other objective organizations should be avoided. Once user profiles are extracted and their opinion is known their posts in the dataset should be labeled accordingly.

B. Semi-supervised Label Prediction (SLP)

In this paper, a general semi-supervised model is proposed which can incorporate different language modeling and classification algorithms. This is in contrast to previously proposed semi-supervised methods which work with a single model and require big documents or additional information or optimization equations. The main purpose of the proposed model is to help in surveying information from social networks, so it is able to work with very small documents and algorithms that provide the best short-text classification results, including deep learning models. Since opinion modeling usually differentiates only a few sides, focus of this framework is on the best performance with a smaller number of classes.

The architecture of the proposed framework shown at **Figure 1** assumes small labeled training data (in thousand samples) and huge unlabeled data (in million samples). Labels are numbered, so they have values $(1, 2, ...L)$ for an L-class problem. Classes can have different meanings, such as reviews scores or opinion on certain topic. The language model and the classification system on top of it are trained on the small labeled training data to get the initial model. This model is used to predict probabilities for labels for each post from unlabeled data. Posts assigned the largest probabilities (at least X times bigger than the next most probable label) were moved from the unlabeled to the training dataset.

This process is repeated and it stops if the amount of posts moved to the training data in an iteration is smaller than PL or the amount of unlabeled data becomes smaller than UL samples. Remaining unlabeled data is assigned a label which had highest probability in the last iteration. This step is optional and it depends on the purpose of modeling.

Enlarging the training dataset based on model-labeled data, as proposed in this paper, allows new contexts to be added to the training dataset with their probable labels. While this is beneficial for models learning from the contexts, adding such model-labeled data also introduces noise into the training dataset since labels are generated with some probability. In our approach, ratio X controls how strict the model should be when accepting new labels. For smaller values of X, more elements will be added to the training dataset and the model will learn faster; however, for larger X more noise will be allowed as well.

The hypothesis explored in our study is that SLP architecture will be the most beneficial for language models that depend on the order of words more than frequencies captured by a bag-of-words based alternative. When working with short texts deep neural networks, which are learning from the contexts and embedding the higher level features and meanings of the language, have shown much better results. Our model iteratively increases the training dataset with well labeled contexts, adding more useful information that can be used for training. It is expected that for models learning from context (e.g. fastText, deep learning models) the benefit added from new examples is bigger than the noise introduced with their most probable labels.

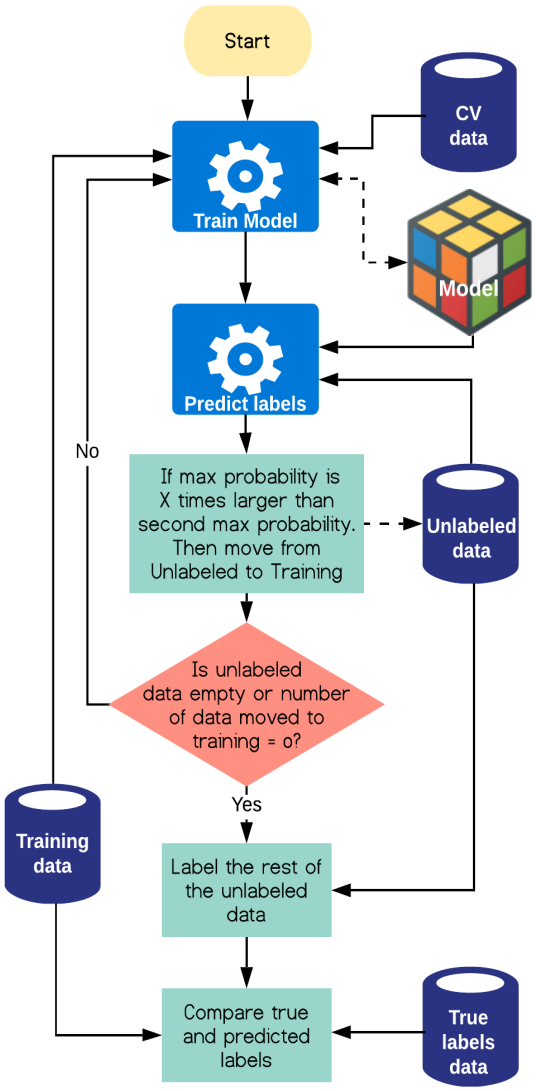

Fig. 1. Architecture of the proposed semi-supervised label prediction (SLP) framework.

C. Handling very small seeds

Some language models (e.g. tfidf+ngrams) perform better for small datasets while others are better with larger datasets

(e.g. vdcnn, fasttext). Since a different model can be trained in each iteration it is possible to combine distinct classification methods to achieve an improved performance. For example, if starting with a small seed, we can use tfidf+ngrams model in the first iteration to add more examples and then use some deep learning model in next iterations.

Another way to work with a very small seed (hundreds of examples) is to use pretrained word-vectors from word2vec or Glove available online as the input of the first step. Since those vectors are created on the general text (usually from wikipedia or news), it is important to add new examples of topic-specific text through the proposed iterative process to get better results. The hypothesis is that by adding new topic-specific contexts, the model is able to adapt and learn new meanings of the words and phrases in the particular domain and therefore achieve better classification accuracy.

D. Exit criteria

Stopping criteria can be changed based on the properties of text, algorithms used for language modeling, and classification or algorithm speed requirements. It is possible to stop the algorithm when the number of iterations NI reaches a certain value or when the amount of labeled posts drops below UP percent of the size of original unlabeled data.

The hypothesis is that data which is not labeled in the iterations contains weak posts that don't express the opinion or posts that contain multiple opinions and would be very hard to annotate even for humans. Both of those cases may be considered as a noise and they don't have to be included in learning the final model.

On the other hand, as cases are very hard to label we may want to leave their classification to the human expert or we may want to gather additional information about them. In case of social networks those posts may contain an image or link that can give a better clue. In case of reviews or comments, additional information can be gained from surrounding comments.

E. Streaming

The proposed model is very useful for streaming data, because it is easy to add new unlabeled and training examples. Additionally, it shows which examples are hard to classify (can't be classified after many iterations), so with a small additional human labeling effort, a lot of new knowledge can be added to the model.

F. Model testing

The model can be tested in two ways: 1) comparison with true labels in case those are known for unlabeled data; 2) testing on a small part that was removed from labeled seed before the training process started. While the second evaluation is meant to be used in real-world practice, it is possible to find labeled datasets with short texts, so if a new combination of language and classification models has to be tested then the first testing option can be used. In this paper we test on huge labeled datasets from [12] and a newly created twitter dataset.

IV. Experiments

There are many variables in the evaluation of the performance of the proposed algorithm, such as language and classification algorithms chosen, datasets characteristics, and the values of the parameters. When designing experiments, components are chosen to minimize the number of experiments while covering different test cases. Since multiple semi-supervised approaches exist for long texts, the main purpose of the proposed model is to work with short text, so its performance is tested only on the short text.

Environment. When text modeling and classification is done with the deep learning model (VDCNN) experiments were conducted on 4 large nodes with 512 GB of DDR4 2400MHz RAM. Each host had two sockets with Intel Xeon E5-2667 v4 3.2GHz processors. Every node featured two NVIDIA Tesla P100 PCIe 12GB GPUs and SSDs as local hard drives. Other experiments were run on a 64-bit processor, Intel Core i7-6700 CPU @ 2.60 GHz with four cores and 16.0 GB RAM.

TABLE I
Large-scale text classification datasets

Dataset	Unlabeled	Data Type	Classes
AG news	128k	Topics	4
Yelp Review Polarity	598k	Reviews	2
Amazon Review Full	3,650k	Sentiment	5
Amazon Review Polarity	4,000k	Sentiment	2
Twitter guns	11,750k	Opinions	2

Datasets. To test different aspects of the proposed model four datasets [12] were chosen with different characteristics. Since this model focuses on opinion, modeling experiments are focused on datasets with small number of classes. Sogou News dataset is avoided because authors don't understand Chinese.

Additionally, the method is applied to a new high-impact problem regarding gun advocacy which we have extracted from twitter. These benchmark datasets cover different classification tasks such as sentiment analysis, opinion modeling, product reviews and news categorization. All published datasets are balanced and so it was not possible to test how class balance influcences the results on these datasets. Real problems are often imbalanced, so the gun advocacy dataset was extracted from twitter to test such a case.

The number of unlabeled examples significantly differ among the used datasets. Moreover, to test the robustness of the model, a different number of classes were considered. A few datasets have two classes, because they differ in size of unlabeled data and text purpose, so those can be tested independently of the number of classes. The number of unlabeled data used within our experiments is reported at Table I. These unlabeled data were extracted by joining the training and testing data from [12] and removing labels for all examples that are not used for training and cross validation in our experiments.

Twitter guns advocacy dataset. This dataset has two classes: gun rights and gun control advocacy. It was downloaded from twitter using an expert-created list of initial hashtags and relevant tweets were selected while noise was reduced according to the previously developed protocol [6]. This dataset contains 11.75 million unlabeled posts with information about the name of a user who posted a message. To start the search, a list of gun rights/control organizations found on Wikipedia was used. More profiles were searched in twitter using hashtags provided by experts for data selection. Additionally, twitter's feature that suggests profiles similar to the current profile was used. Such profiles were recursively explored, and if some profile was closely related to the particular class, all of its posts were labeled accordingly.

A profile was considered only if its description clearly stated that the goal aligns with one of the classes. With this approach, 171 profiles were selected, 88 labeled as gun rights advocacy and 83 labeled as gun control advocacy groups. They had in total 8,448 posts in the downloaded dataset, out of which 7,782 posts were in the gun control class, i. e. downloaded data is imbalanced.

This dataset was used to simulate an experiment that can be taken by a researcher or an opinion research center. It tests the performance of the SLP model in an unbalanced case. Beside modeling public opinion, this approach can be used for removing posts from classes that are not of interest. This, and related problems, usually produces heavily imbalanced datasets which is why they were examined in this study.

Parameters settings. In this model, the main parameters are **ratio** and **seed size**. The proposed algorithm repeats training and prediction phases until it is not able to label any post or unlabeled data is empty. However, different stopping criteria can be introduced to finish the iterative labeling process earlier. A few options are possible, such as $PL \in \mathbb{N}_0$ - amount of examples added to the training dataset in the current iteration, $UL \in \mathbb{N}_0$ - number of examples left in the unlabeled dataset, $NI \in \mathbb{N}_0$ - number of iterations passed and $UP \in \mathbb{N}_0$ - percent of unlabeled data that is still unlabeled. In case when training component works with batches, newly labeled data should contain at least a few batches, so if used, PL and UL should be multipliers of the size of the batch. Those variables mainly influence the run time of the algorithm. They can also be used to extract a certain percent or amount of unlabeled data which are very hard to label. They are not required and so will not be discussed further.

Ratio X decides if the probability of a predicted label is sufficiently large such that an automatically annotated example can be moved to the training dataset. In the prediction step, each label is assigned a probability, then the two biggest values are compared for each post. If the ratio between them is larger than X, the post is moved into training data for the next iteration. In conducted experiments considered ratios were 2, 3 and 4. When the training model performed best for ratio 2 and accuracy becomes lower with bigger ratio, additional experiments were run with ratios 1.2 and 1.5.

Seed size is the number of labeled examples used for the training and cross validation process (if required by the training method). Two seed sizes were tested in details ($6.7K$ and $67K$), out of which $5K/50K$ were used for training and $1.7K/17K$ were used for cross validation. Other considered options for seed sizes were 500 and $500K$. Labeling 500K of data requires great labeling effort which is not in focus of this paper. Existing classification models already perform well on this dataset size. However, in recent machine translation work [28] it is shown that continuous training brings additional benefit even with huge datasets. Since 500 samples of short-text contain too little text and many algorithms can't perform well with so little data, we added pretrained GloVe vectors from [29] [2] as input to the first iteration classification.

Models used for training step. SLP is a framework designed to incorporate different language models and classification algorithms. Existing language modeling options can be divided into three main types: 1) traditional models based on word-document matrix; 2) deep learning models; and 3) word2vec based models and GloVe. One algorithm from each group was used for the training step of the proposed model: 1) tfidf of character n-grams, where $n \in \{1, 2\}$; 2) very deep convolutional neural network (VDCNN) with 9 layers and k-max-pooling as in [7]; and 3) fastText classification from [30]. Since tfidf+ngram models the language only, logistic regression (trained using stohastic gradient descent) was used on top of it to classify the documents. Additionally, because of their complementary properties, tfidf+ngram and fastText were combined. In the first iteration tfidf+ngram was used because it performs well on small data. In all other iterations fastText was used because training size became large enough, so benefits of this algorithm could be used.

Comparison models. The proposed model was compared to classification models: tfidf+ngram, VDCNN, and fastText.

Time performance. The proposed model has the same time complexity as a model that is used for its training. Label prediction time is always much shorter than training time. The number of iterations can be considered as a constant because it is in range (2-30) depending on the dataset characteristics, training algorithms used, and stopping criteria. If deep learning models are used, the model doesn't have to be trained on the same examples multiple times. Newly labeled data can be used for further training of the model as if a new batch was added. Therefore, in that case, the training time of SLP with a deep learning classification component is close to the time of the classification algorithm only.

V. Results

Results are shown in Table II and III for seeds size 5K and 50K, respectively. Three text classification models from literature (VDCNN, ngrams+tfidf with logistic regression on top and fastText) were compared with the results of the proposed model (SLP). Training step of SLP was implemented in four different ways: VDCNN, ngrams+tfidf with logistic regression

[2]Pretrained 300-dimensional vectors of 16B tokens of wikipedia text: https://dl.fbaipublicfiles.com/fasttext/vectors-english/wiki-news-300d-1M.vec.zip.

TABLE II
LABEL PREDICTION ACCURACY (LP) FOR SEED SIZE = 5,000. FOR EACH VALUE OF THE PROPOSED MODEL, THE BEST RATIO FOR WHICH THIS VALUE WAS OBTAINED IS GIVEN IN BRACKETS.

Model	AG	Amz. F.	Amz P.	Yelp P.	Twiter
VDCNN	0.375	0.286	0.683	0.790	0.921
ngrams+tfidf	0.729	0.270	0.744	0.776	0.954
fastText	0.336	0.201	0.500	0.500	0.920
SLP(VDCNN)	0.645 (4)	0.285 (2)	0.758 (3)	0.521 (4)	**0.965 (4)**
SLP(ngrams+tfidf)	0.721 (1.2)	0.311 (1.2)	0.731 (1.2)	0.807 (1.2)	0.927 (1.2)
SLP(fastText)	**0.855 (2)**	**0.409 (2)**	**0.817 (2)**	**0.810 (2)**	0.923 (4)
SLP(fT+ngrams+tfidf)	0.797 (3)	0.354 (1.2)	0.771 (3)	0.744 (1.2)	0.951 (1.5)

TABLE III
LABEL PREDICTION ACCURACY (LP) FOR SEED SIZE = 50,000. FOR EACH VALUE OF PROPOSED MODEL THE BEST RATIO FOR WHICH THIS VALUE WAS ACHIEVED IS GIVEN IN BRACKETS.

Model	AG	Amz. F.	Amz P.	Yelp P.
VDCNN	0.869	0.458	0.862	**0.908**
ngrams+tfidf	0.757	0.388	0.749	0.805
fastText	0.853	0.387	0.809	0.501
SLP(VDCNN)	**0.884(3)**	**0.482(2)**	**0.876 (4)**	**0.908(3)**
SLP(ngrams+tfidf)	0.750(4)	0.372(4)	0.731(1.2)	0.809(4)
SLP(fastText)	0.871(2)	0.431(2)	0.840 (2)	0.863(2)
SLP(fT+ngrams+tfidf)	0.861(4)	0.398(1.2)	0.816(4)	0.808(1.2)

on top, fastText, and a combination of ngrams+tfidf+fastText. Since true labels for the huge unlabeled datasets, taken from the literature, are known, accuracy is calculated by comparing true and predicted labels of that dataset. Due to the vast amounts of examples variability of accurate results achieved when an experiment was repeated was insignificant. Additionally, all of the models (including previously published classification algorithms) were tested using 1.7K/17K labeled samples from each dataset. However, because of the small number of test samples (especially in 1.7K test dataset), those results were more variable. When the model was trained with a seed of 5K labeled documents and tested with 1.7K documents results varied from 0.1% to 5%. When the model was trained with 5K and tested with 17K examples results varied between 0.1% and 1.5%. Therefore, accuracy reported in Tables II and III was calculated as a percent of accurately labeled examples from unlabeled dataset, because this accuracy is stable. In the case of twitter data, where correct labels of unlabeled data are not available, accuracy was calculated as an average of testing results achieved by multiple experiments.

For seed size 5K the best results were obtained using SLP trained with fastText for four datasets, while SLP trained with VDCNN was the best for the last two datasets. In the four datasets where SLP trained with fastText was the best among the algorithms, that was achieved for ratio $X = 2$.

It is interesting to make comparisons between the original algorithm and SLP that uses the same algorithm for the training step. As hypothesized, SLP with ngrams+tfidf was rarely better than the original ngrams+tfidf because this is a bag-of-words algorithm which can capture frequencies of words in texts quite well with small training dataset and doesn't benefit from the context. Therefore, contexts added with additional automatically labeled samples did not bring much new information, but they introduced noise making SLP trained with ngrams+tfidf perform worse in four datasets for both seed sizes. On the other side, SLP trained with VDCNN or fastText was better than the original VDCNN and fastText for most of the datasets for both seed sizes. Those algorithms rely on contexts, so adding new training samples improved their performance despite the noise in added labels.

For a smaller seed size (5K) SLP trained with ngrams+tfidf obtained the best results for ratio X = 1.2, while SLP trained with VDCNN was the best mainly with higher ratios and SLP trained with fastText most often achieved the best results with ratio 2.

We made a few experiments with datasets with 10 (Yahoo Answers) and 14 (DBpedia) different classes. Performance of our framework on them was often only a bit better than the random choice. With a small number of examples and a big number of classes, there are very few labeled examples per class. Additionally, text is short and noisy, so even the best classifiers with a lot of data have big error rates. Moreover, DBpedia dataset consists of entity categories and text is formal description of entity. Most of the words are therefore used only once to describe relevant information about that entity, while other words are used very often, i.e., words have power-law distribution. Neither of those characteristics help in distinguishing the classes of texts when a small amount of labeled samples is available, which is why the algorithm performed poorly.

The accuracy of classification was highly dependable on the number of classes in general. The best accuracy for Amazon Full dataset was 0.409 and 0.482 for 5K and 50K seeds sizes, respectively. This performance was almost half as large as the accuracy achieved on the Amazon Polarity dataset, even though texts used were the same. Therefore, an increasing number of classes from two to five profoundly influences the accuracy of those algorithms.

When it comes to the influence of the size of unlabeled data on accuracy, performance was better when the unlabeled amount of data was smaller, i.e., when the percentage of labeled data in the whole dataset was more significant.

In Figure 2 influence of the ratio X on the accuracy of SLP

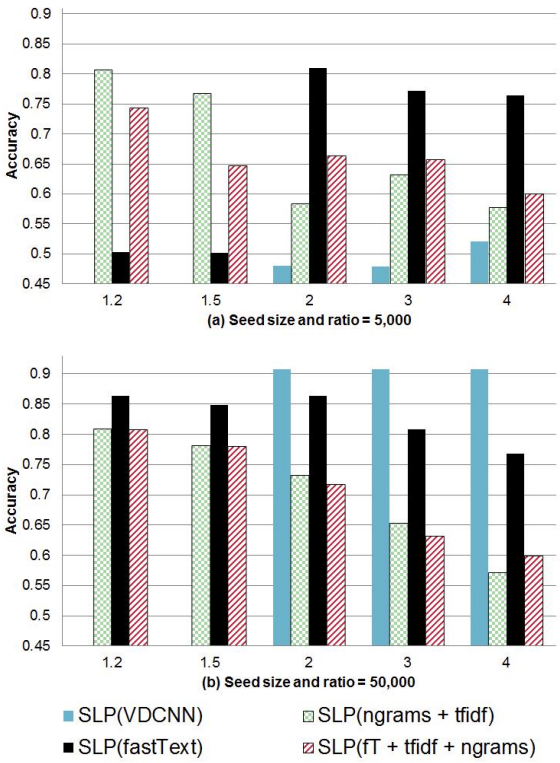

Fig. 2. Accuracy of SLP model trained with multiple language modeling algorithms and different ratios for (a) seed size = 5,000 and (b) seed size = 50,000 for Yelp Review Polarity dataset

is shown for the Yelp Review Polarity dataset. Higher ratio value negatively influenced SLP trained with tfidf+ngrams and SLP trained with tfidf+ngrams+fastText. On the other side, fastText reached its maximum at ratio 2 for all the examples, and then it slowly dropped. When other parameters are kept constant, seed size impacted accuracy positively in most cases.

To check the hypothesis that our model can be used on very small datasets (few hundred examples) when pretrained vector embeddings were added as input, we tested all datasets on 500 examples using fastText algorithm (which sufferes the most on the very small data). Adding pretrained vectors embeddings to the fastText classification, improved results 8-27%. This accuracy is additionally improved when SLP is trained with fastText initialized with pretrained vector embeedings. Influence of initialization is the biggest on AG news data which is probably due to similarity between news texts and wikipedia data being the biggest comparing to the other datasets. Improvement that SLP brings depends more on text type than on the seed size.

In order to support the hypothesis that examples, which were left unlabeled in iterations of SLP, were confusing or contained weak messages, those examples were examined. For example, in Amazon Full dataset most of the reviews that were not labeled held different opinions inside them (e.g. people who think product is not good, but its cheap, so it serves the purpose, or people who loved product but some event changed their opinion). Examples are shown below:

"The tacos shells were mainly broken. shells are too thin for transportation. the shells are very good. mainly use for dipping."

"grass grows quickly and its cool at first, but with no sun, eventually molds, roots invade the bowl,and my cats didn't catch on to it.... pfft"

If last 5000 hardest examples to label are discarded, SLP model accuracy increases by 3-5%.

Finally, even though the accuracy of all models was high for the twitter dataset, it was not the best performance indicator since the dataset is imbalanced, which is why F1 score was also measured. The best F1 score (0.675) was achieved by SLP trained with a combination of ngrams+tfidf+fastText, while the worst score was obtained by fastText classification. While the original VDCNN and tfidf algorithms can incorporate information about imbalanced classes through their settings, fastText works poorly in this case. Therefore, SLP which uses tfidf and VDCNN achieved good F1 score, while SLP with fastText had low F1 value.

VI. Conclusions

In this study, a semi-supervised label prediction (SLP) model is proposed to label a large number of documents based on a small number of labeled samples (seed). A training component can be chosen based on the problem and the amount of labeled data. The model performance is compared to predictions made by state-of-the-art classification models.

The obtained results provide evidence that our model outperforms state-of-the-art classification models in labeling vast amounts of unlabeled data based on the small seed, except when the number of categories is substantial. It is shown that the choice of training component is essential and related to the dataset characteristics.

In future work, we would like to test performance with other classification models, such as HAN and performance of the model when the first step is initialized with word embeddings, especially on datasets with a lot of classes. We expect performance improvement to be reversely proportional to the difference in topic and text style between current dataset and datasets which were used to create word embeddings.

VII. Acknowledgements

This research was supported in part by the NSF grants IIS-8142183, CNS-1625061 and Pennsylvania Department of Health CURE Health Data Science Research Project.

References

[1] E. Otte and R. Rousseau, "Social network analysis: A powerful strategy, also for the information sciences," *Journal of information Science*, pp. 441–453, 2002.

[2] A. Java, X. Song, T. Finin, and B. Tseng, "Why we twitter: Understanding microblogging usage and communities," in *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, ACM, 2007, pp. 56–65.

[3] D. J. Watts and P. S. Dodds, "Influentials, networks, and public opinion formation," *Journal of consumer research*, pp. 441–458, 2007.

[4] A. Pak and P. Paroubek, "Twitter as a corpus for sentiment analysis and opinion mining.," in *LREc*, vol. 10, 2010, pp. 1320–1326.

[5] C. J. Glynn, *Public opinion*. Routledge, 2018.

[6] C. Buntain, E. McGrath, and B. Behlendorf, "Sampling social media: Supporting information retrieval from microblog data resellers with text, network, and spatial analysis," in *Proceedings of the 51st Hawaii International Conference on System Sciences*, 2018.

[7] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," *ArXiv preprint arXiv:1606.01781*, 2016.

[8] M. Steinbach, G. Karypis, V. Kumar, *et al.*, "A comparison of document clustering techniques," in *KDD workshop on text mining*, 2000.

[9] A. Rangrej, S. Kulkarni, and A. V. Tendulkar, "Comparative study of clustering techniques for short text documents," in *Proceedings of the 20th international conference companion on World wide web*, ACM, 2011, pp. 111–112.

[10] J. Yin and J. Wang, "A dirichlet multinomial mixture model-based approach for short text clustering," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2014, pp. 233–242.

[11] J. Xu, P. Wang, G. Tian, B. Xu, J. Zhao, F. Wang, and H. Hao, "Short text clustering via convolutional neural networks.," in *VS@ HLT-NAACL*, 2015, pp. 62–69.

[12] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.

[13] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *ArXiv preprint arXiv:1301.3781*, 2013.

[14] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162.

[15] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International Conference on Machine Learning*, 2014, pp. 1188–1196.

[16] P. Zhou, Z. Qi, S. Zheng, J. Xu, H. Bao, and B. Xu, "Text classification improved by integrating bidirectional lstm with two-dimensional max pooling," *ArXiv preprint arXiv:1611.06639*, 2016.

[17] P. Liu, X. Qiu, and X. Huang, "Recurrent neural network for text classification with multi-task learning," *ArXiv preprint arXiv:1605.05101*, 2016.

[18] Y. Xiao and K. Cho, "Efficient character-level document classification by combining convolution and recurrent layers," *ArXiv preprint arXiv:1602.00367*, 2016.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[20] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1480–1489.

[21] L. Wu, A. Fisch, S. Chopra, K. Adams, A. Bordes, and J. Weston, "Starspace: Embed all the things!" *ArXiv preprint arXiv:1709.03856*, 2017.

[22] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the eleventh annual conference on Computational learning theory*, ACM, 1998, pp. 92–100.

[23] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," *Machine learning*, pp. 103–134, 2000.

[24] S Dharmadhikari, M. Ingle, and P. Kulkarni, "Analysis of semi supervised learning methods towards multi label text classification," *International Journal of Computer Applications*, 2012.

[25] B. Liu, X. Li, W. S. Lee, and P. S. Yu, "Text classification by labeling words," in *AAAI*, vol. 4, 2004, pp. 425–430.

[26] V. Sindhwani and P. Melville, "Document-word co-regularization for semi-supervised sentiment analysis," in *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, IEEE, 2008, pp. 1025–1030.

[27] C. Llewellyn, L. Cram, and A. Favero, "Avoiding the drunkard's search: Investigating collection strategies for building a twitter dataset," in *2016 IEEE/ACM Joint Conference on Digital Libraries (JCDL)*, IEEE, 2016, pp. 205–206.

[28] B. Thompson, H. Khayrallah, A. Anastasopoulos, A. McCarthy, K. Duh, R. Marvin, P. McNamee, J. Gwinnup, T. Anderson, and P. Koehn, "Freezing subnetworks to analyze domain adaptation in neural machine translation," *ArXiv preprint arXiv:1809.05218*, 2018.

[29] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin, "Advances in pre-training distributed word representations," in *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[30] E. Grave, T. Mikolov, A. Joulin, and P. Bojanowski, "Bag of tricks for efficient text classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL*, 2017, pp. 3–7.