Open Classification of Text Document Topics

Qian Yu
UC Berkeley MIDS
q̃ianyu@ischool.berkeley

Varadarajan Srinivasan
UC Berkeley MIDS
ṽaradarajan@ischool.berkeley

Leslie Teo
UC Berkeley MIDS
l̃teo01@ischool.berkeley

Abstract

Due to the dynamic nature of the online text, new online documents may not belong to any of the previously defined training classes. Deep Open classification (Shu, Xu, and Liu, 2017) is a new deep learning based approach that presents a solution to this challenge. The architecture consist of a CNN architecture with a 1-vs-Rest output layer.

We leverage the underlying method laid out in Xu and Liu, 2017, but modify it to explore clustering algorithms in the output layer to determine open class documents. We compare our experiment with the results reported by the DOC reference paper (Shu, Xu, and Liu, 2017). Our results show that, at least for the data and tuning we are able to perform, a 1-vs-Rest approach still does better than clustering algorithms in identifying the "unseen" class.[1]

1 Credits

This project is based on the seminal paper on open classification titled: DOC: Deep Open Classification of Text Documents (Shu, Xu, and Liu, 2017). We also referred to other papers on lifelong machine learning (Chen and Liu. 2016), convolutional neural network for sentence classification (Kim, 2014), paragraph vector (Le and Mikolov, 2014) and task clustering (Thrun and O'Sullivan, 1996)

We are also really grateful to Ian Tenney for reviewing our recommendations, shaping the proposal and mentoring us along the way.

[1]`https://github.com/qianyu88/W266_project_submission`

2 Introduction

News websites have a need to identify new topic classes as they continuously receive streams of new data. A natural language processing model can be used to quickly identify whether an incoming news feed is related to an existing set of topics or a new topic. A supervised text classification model can be trained to learn and classify documents based on topics or genres with good labeled training data. However, in the web 2.0 world, new content is constantly being generated by social media, news articles, and blogs. Due to the dynamic nature of this content, a new incoming document may not belong to any previously "known" classes but rather a new but unseen one. The key assumption of supervised learning of predicting based on what has been observed before at inference time is therefore violated.

One approach to identifying new topic classes is called open world classification (Fei and Liu, 2016) in which an 1 vs. Rest classifier is trained to detect an unseen class. Open classification is also part of a new machine learning paradigm called Lifelong Machine Learning (LML) (Chen and Liu, 2014a). It is particularly valuable in learning the abundant and multifarious information from the web. In the natural language learning setting, open world classification can be not only be used to filter unwanted documents but also discover new categories. Open world classification has several real world applications, namely, (1) identifying new topics, genres in social media e.g. new twitter topics, news or facebook trends (2) Filtering email or other text documents where topics may grow or change over a period of time and (3) Online learning (Thrun and O'Sullivan, 1996).

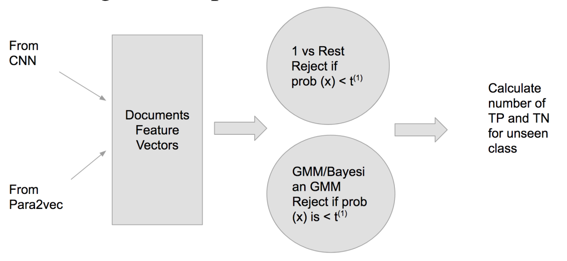
3 Background

Our implementation of open world classification builds on the approached proposed in DOC paper (Shu, Xu, and Liu, 2017). As suggested in that paper, we also used a *Convolutional Neural Network* (CNN) architecture due to CNN's performance and efficiency gains on sentence classification (Kim, 2014) tasks. In this architecture, a 1-vs-Rest output layer with *m* sigmoid functions is used for open classification where *m* is the number of "known" classes. The prediction of sigmoid function is reinterpreted at the testing time to determine the unseen open class. A document is classified as belonging to the the open (or unseen) class if its sigmoid probabilities are less than thresholds of all labeled classes. We built on the DOC paper architecture by (a) modifying the method by which the threshold for marking a document as part of an "unseen" class is determined. In our approach we use a validation dataset to estimate a percentile threshold that maximizes the unseen class F1 score while also ensuring unseen class predicted volume is in line with actual unseen class volume in validation set, and (b) as an enhancement to Shu, Xu and Liu's 1 vs.Rest approach, we applied unsupervised clustering methods in the output layer to predict an open class document.

Using a clustering method for online learning is a well-known practice. Particularly, task clustering (Thrun and O'Sullivan, 1996) is an old concept but is based on a similar idea to lifelong learning. This idea can also be applied to open classification problem. In task clustering, when a new task arrives, algorithm first selects the most similar cluster then uses the distance function of the cluster for classification (Trun, 1996b). Concretely, we can take the trained feature vectors of documents from the language model and use them as input parameters of a unsupervised clustering analysis. Using an outlier detection approach, we define threshold for labeled clusters' probability distribution. If a new document is detected as an outlier of all clusters, it belongs to an "unseen" class.

We experimented 2 different clustering methods: Gaussian Mixture model (GMM) and Bayesian Gaussian Mixture model (BGMM) in particular the Infinite Dirichlet Process (IDP). BGMM is a variant of the GMM with variational inference where the algorithm maximizes a lower bound on model evidence with consideration of priors. IDP is a prior probability distribution on clusterings with an infinite, unbounded, number of partitions. The model fits with the nature of open classification where we can have infinite number of unseen classes. *Figure 1* shows a high-level view our open classification work flow.

Figure 1: Open Classification Flow


From CNN
From Para2vec
Documents Feature Vectors
1 vs Rest Reject if prob (x) < $t^{(1)}$
GMM/Bayesian GMM Reject if prob (x) is < $t^{(1)}$
Calculate number of TP and TN for unseen class

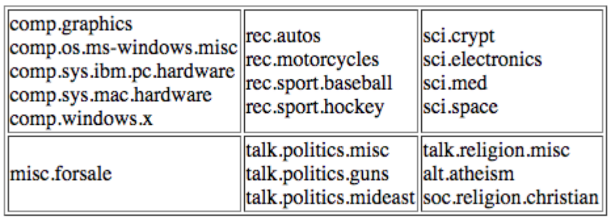
Note: (1) Threshold t was learnt on validation set by maximizing F1 score of unseen class and ensuring volume of predicted unseen class was comparable to actual unseen class in validation set

4 Methods

4.1 Data Set

We used **20 Newsgroups** (Rennie, 2008) data set for our experiment. The data set contains 20 non-overlapping classes (*Figure 2*) in newsgroup topics. Topics were divided across 6 broader themes (politics, religion, recreation, computer, science and for-sale). Each class has around 1000 samples.

Figure 2: Open Classification Flow



comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

4.2 Paragraph Vector Model As Baseline

As a baseline model, we used paragraph vector model (Le and Mikolov, 2014) for our experiment.(*Figure 3*) The paragraph vector model produces a fixed-length feature representations of a paragraph from variable-length pieces of texts. Compared to Bag of Words (BOW) model, it provides a dense feature vector representation of documents, capturing ordering and semantics of words which is similar to a CNN model. We used the Gensim library's Doc2vec implementation of paragraph vector and configured the API to use the distributed memory model proposed in the [Le, Mikolov] paper. DM model is inspired by the methods for learning the word vectors. We learn the paragraph vectors by running the training data through distributed memory model (DM) in Doc2Vec implementation and at inference step, we infer the vector representation for the learned dataset. Ideally, we should have used a separate sample for training the paragraph vectors and a holdout set for inference.Unfortunately due to small sample size of the newsgroup dataset, we ended up using the same training set for inferring the paragraph vectors. The paragraph vector model uses the max vocab size of 20000 and outputs a feature vector size of 450 to align with feature size of the CNN model. Paragraph vectors also address some of the key weaknesses of bag-of-words models. They inherit an important property of the word vectors: the semantics of the words, for e.g., the word 'powerful' is closer to 'strong' than to 'Paris'.

4.3 CNN Model Architecture

As our primary model we developed a Convolutional neural network architecture. (*Figure 4*) Based on the recent research on document and sentence classification using CNN (kim, 2014; Zhang and Wallace, 2016), it has been reported that CNN offers an excellent performance in sentence classification tasks compared to other state of the art techniques such as RNN. A big argument for CNNs is that they are fast. Convolutions are a central part of computer graphics and implemented on a hardware level on GPUs. Compared to something like n-grams, CNNs are also efficient in terms of representation. With a large vocabulary, computing anything more than 3-grams can quickly become expensive. Even Google doesnt provide anything beyond 5-grams. Convolutional Filters learn good representations automatically, without needing to represent the whole vocabulary.

We compared the approaches of training the word embedding layer along with the CNN model and used Google's pre-trained word2vec as embedding layer. Google pre-trained word2vec offered better performance than training word2vec on newsgroups data, therefore we used Google pre-trained word2vec as the CNN embedding layer for all our experiments.

Each document in the data set is padded or cut into a fix length in words. We use 500 words length which is at the median length of all document in the dataset. This number was chosen to balance training speed and information loss because the distribution of the documents in the dataset is following the power-law. Each document is transformed into an 500×300 dense matrix with embedding look up table. The CNN internal dimension is mirror to the DOC architecture (Shu, Xu, and Liu, 2017) where 3 regions of $[3, 4, 5]$ and 150 filters was used. The maxpooling layer output which is used for open classification analysis has the feature size of 450.

4.4 Open Classification Methods

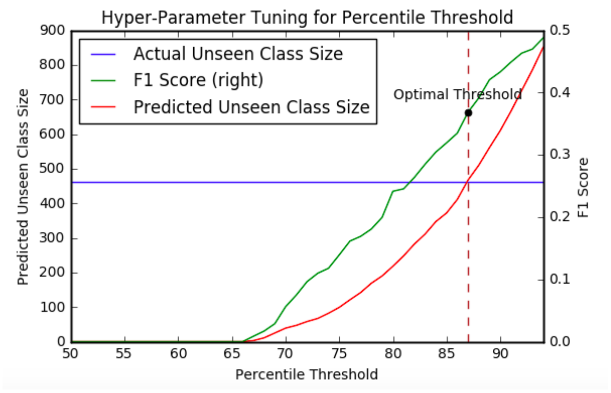
The mechanics of the open classification analysis is the following: We train the language models using paragraph vector or CNN. We then extract feature vectors from the language model. Lastly we run open classification experiments on the extracted features using 1-vs-Rest and Unsupervised Clustering.

1-vs-Rest is the baseline method we used for open classification. We hold out one or more classes from the training data and then at testing time we then mix them back into our sample. To determine if a new document is "unseen", we first train a model & calibrate the probability of 1-vs-rest predicted probability for labeled classes. We then predict the probability of all documents in test classes using that model. We then compare the probability of the test sample to the probability distribution of the labeled classes using a threshold. If its probability is smaller than the threshold for all labeled classes, then the new sample is classified as 'unseen'. We use F1 score of 'unseen' class as the measure to evaluate effectiveness of our predictions. 1-vs-Rest classifier requires a baseline classification algorithm and we tried both logistic regression and SVM 1-vs-rest classifiers. The performance difference was less than 1%. Our reported results is based on multinomial logistic regression 1-vs-rest classifier.

For clustering methods, we first performed dimensionality reduction because the scikit-learn library we used for clustering analysis could not handle higher dimensional embedding size (of 450). We chose Latent Semantic Analysis method as it enables discovery of latent patterns in the data. We used SVD and LSA normalizer to collapse the dimension of (CNN/Paragraph2Vec) trained vectors (dimension ($D \times 450$)) to a latent dimensions of ($D \times 20$), where D is number of documents. We then used Gaussian Mixture Models to fit "m" gaussians on the LSA transformed data for "m" seen classes. We fixed the number of components equal to the number of known classes and used Bayes Information Criteria to select appropriate covariance matrix.

As last step, to predict the unseen class, we developed an approach similar to outlier detection task. As mentioned previously for 1-vs-rest method, we hold out one or more classes from the training set and mix them back during the validation and test phase. Once the GMM model is trained on ”known” classes, we then predict the probabilities on a validation set which comprises of both known as well previously unseen classes. Hyper parameter ”percentile threshold” is varied to maximize the F1 score of the unseen class in validation set. As a high value of percentile threshold will result in marking more documents as ”Unseen” and vice-versa, we had to be judicious about the choice of threshold, so we added another criteria of creating an ’unseen’ class size that was equivalent to true unseen class size in validation data (Figure 5). This combined criteria of F1 score and unseen class size was used to derive optimal threshold which was then used in test dataset to predict the test set metrics. We followed this same approach for both GMM and Infinite Dirichlet Process (IDP) clustering methods.

Figure 3: Percent Threshold Tuning on Validation Set



5 Results and Discussion

5.1 Test Metrics

We used 64% (Training), 16% (Validation) and 20%(Test) data split and random shuffle for running these experiments. We held out one or more classes at training time and added them back during validation and testing phase. We also tested with holding back 1, 2, and 3 unseen classes, however for unseen class detection task all unseen classes were bundled together into one set, so for example, when we held back 3 unseen classes, we pooled them together as one large unseen class instead of predicting each of the 3 unseen classes independently. (Note: Initially, we set an ambitious target of training all 20 possible classes but we had to scale back due to the time and computing cost of training CNN models.)

We used a weighted average of precision and recall to compute F1-score over the ”unseen” class for evaluation. In other words, for this project we are focusing on how well our model is predicting the ”unseen” samples only without considering the performance of all classes.

5.2 Results and Error Analysis

Model	Open Method	5+1	5+2	5+3
P2vec	1-vs-Rest	39.0%	51.0%	66.0%
P2vec	GM	12.5%	29.2%	35.2%
P2vec	IDPs	13.8%	28.5%	37.6%
CNN	1-vs-Rest	27.0%	34.0%	41.0%
CNN	GM	13.5%	31.9%	38.8%
CNN	IDP	10.8%	27.2%	38.0%

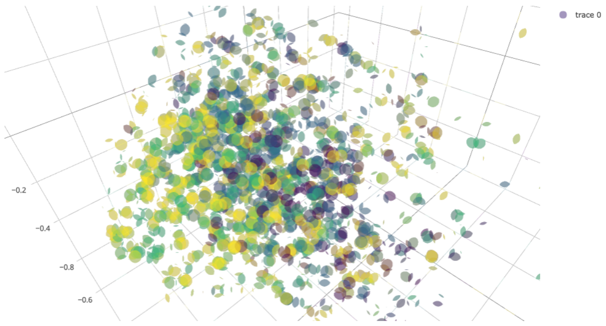
Table 1: F1-Score for Unseen Class for 5 Seen + 1, 2, 3 Unseen Experiments

While we were not able to run our test over all 20 groups, the results for our 5+1, 5+2, and 5+3 are quite instructive.

First, the paragraph to vector model did better than the CNN model when using 1-vs-rest approach and is on-par with the CNN model when using clustering approaches for our open classification experiments. We attribute this largely to a hyper-parameter tuning problem. Before running the open classification tasks, we trained the CNN model using a soft-max output layer and compared its performance with the paragraph vector model for closed class classification. Due to the limitation of computing resource and time, we could not tune the CNN model hyper-parameters to outperform paragraph vector in closed classification tasks. As a baseline when we used 5 labeled classes from the 20 newsgroup data for closed classification, the paragraph vector model based classifier outperformed the CNN model by 14% points (85% (Pvec) vs. 71% (CNN) in accuracy score).

Secondly, the 1-vs-Rest open classification approach performed better than clustering methods. We believe that the following 2 reasons contributed to this performance difference: (1) Reducing the dimensionality of pre-trained feature Vectors into latent dimensions (using LSA) potentially resulted in information loss. Though LSA transformation captured 70% of the variance in those dimensions, the classes were clustered too close to each other to generate clean clusters. To illustrate that we embedded the LSA transformed data using (t-SNE) distributed stochastic neighbors embedding to visualize these clusters. As figure 6 shows, the data points across classes are co-mingled and clustered too close to each other.

Figure 4: 3D plot of Test set by top 3 Latent Dimensions



examining the errors, we observe some patterns in the data which confirms this spatial closeness/co-mingling of classes hypothesis. The model struggled to separate out/cluster certain ambiguous documents accurately i.e. a document that belongs to one class such as Sci.Medicine, but has information that can lead to mis-classification in other classes such as talk.politics. For example, test sample 31 is an article under science medicine genre, and this document refers to funneling of federal funds allocated to health care to support defense expenditure, which has references to ”politicians”. This news article was supposed to be part of sci.med which was an ’unseen’ test class, but the gmm model predicted this under talk.politics. Losing the contextual information from lower dimensional embedding is potentially placing heavier weight on the remaining word ’politicians’, which is referenced throughout the article resulting in this mis-classification. (2) the GMM clustering algorithm allows for a very rich and complex clustering of the data. Our sample size was quite small (with only a few thousand documents) which resulted in skewed probability distributions which made selection of outliers based on those threshold to be highly sensitive.

Finally, We do find that both paragraph vector and CNN performed better as the number of unseen classes increased which aligns with our hypothesis that larger samples in unseen classes can lead to richer probability distributions resulting in much more reliable thresholds for unseen class predictions. This hypothesis, however, does have to be confirmed with further experiments on a different dataset of larger sample sizes.

6 Conclusions and Future Work

In summary, our results indicate that a 1-vs-Rest approach generally does better than clustering approaches in identifying unseen classes. Without significant tuning, a paragraph vector model does better than CNN model using 1-vs-rest method and is on-par with CNN model using clustering methods. We do see better results in both methods as the number of unseen classes increase.

Our potential next steps are to: A) fine tune CNN model using larger data sets to optimize classification performance before perform open classification tasks; B) Study the trade-off between ”unseen” sample prediction accuracy and labeled sample prediction accuracy in open classification setting; and C) Explore application of open classification in an online learning setting.

References

Lei Shu, Hu Xu, and Bing Liu. 2017. *DOC: Deep Open Classification of Text Documents.*

Zhiyuan Chen and Bing Liu. 2016. *Lifelong Machine Learning.*

Yoon Kim. 2014. *Convolutional neural networks for sentence classification .*

Ye Zhang and Byron C. Wallace. 2016. *A Sensitivity Analysis of (and Practitioners Guide to) Convolutional Neural Networks for Sentence Classification*

Zhiyuan Chen and Bing Liu. 2014. *Mining topics in documents: standing on the shoulders of big data.*

Quoc Le and Tomas Mikolov. 2014. *Distributed Representations of Sentences and Documents*

Sebastian Thrun and Joseph OSullivan. 2014. *Learning More From Less Data: Experiments With Lifelong Robot Learning*