Text Categorization using Gaussian Mixture Models

Maria Apostolidou
Electrical and Computer Engineering
Technical University of Crete
Chania, Greece
Email: mapostolidou@isc.tuc.gr

Nikolaos Kyparissas
Electrical and Computer Engineering
Technical University of Crete
Chania, Greece
Email: nkyparissas@isc.tuc.gr

***Abstract*—Text categorization plays a very important role in the development of natural language processing methods that promise to enable automated analysis of digital corpora as well as further optimizations in the human-machine communication interface. This work demonstrates methods of single-labeled text classification using Gaussian mixture models. Classifiers are trained using various labeled data and their performance is analyzed in experimental sequences employing various parameter sets.**

I. Introduction

The wide-spread availability of Internet access world-wide in combination with the development of cheap smartphones and the advent of social media caused a sharp increase in the availability of digital text corpora. Extracting information and classifying such corpora presents industrial, consumer as well as academic interest. Natural language processing (NLP) focuses on the development of methods that will enable not only automated analysis of the available corpora, but will also allow for an ever more natural communication interface to be developed between the human user and *smart* devices.

An important method of NLP is text categorization (TC) which concerns the classification of corpora components. This classification can be implemented with various degrees of granularity, from single characters up to whole paragraphs and complete texts and has a variety of applications, e.g. from sentiment analysis to summarization and discourse parsing.

In this project we implemented a system to solve a single-labeled text classification problem. More specifically, we designed a call routing system to categorize customers' phone calls to specialized phone operators. We used labeled, text converted sentences for supervised machine learning. We applied the term frequency - inverse document frequency (TF-IDF) weighting scheme and latent semantic analysis (LSA) to reduce the dimensionality of the problem. We then estimated the distribution of the features with the use of Gaussian mixture models (GMMs) and followed the maximum likelihood (ML) rule to categorize the testing samples.

The paper is structured as follows: we first present a short survey of significant state-of-the-art methods in Section II and then provide an elaborate description of the steps that were used in our approach to the problem in Section III. The process that we used to test our implementation is presented in Section IV and we summarize with the testing results and suggestions for further improvements in Section V.

II. State of the Art

A. Naive Bayes

The naive Bayes classification is widely used in TC [1]. Although the name suggests a simple approach to the problem, the results of naive Bayes classification are competitive to many complex methods of classification. Its computational simplicity along with the relatively high accuracy results make it an excellent choice for solving TC problems. Given a corpus D containing r documents $D = \{\vec{d_1}, ..., \vec{d_r}\}$ and a set of q categories $C = c_1, ...c_q$ the classification is accomplished according to the Bayes rule:

$$P(c_k|\vec{d_i}) = \frac{P(c_k)P(\vec{d_i}|c_k)}{P(\vec{d_i})}. \tag{1}$$

Under the assumption that the words of a document are independent from one another, the probability $P(\vec{d_i}|c_k)$ can be simplified as the product of probabilities:

$$P(\vec{d_i}|c_k) = \prod_{j=1}^{|T|} P(w_{ji}|c_k) \tag{2}$$

where $\vec{d_i} = \{w_{1i}, ..., w_{|T|i}\}$, $|T|$ is the number of terms in document $\vec{d_i}$ and w_{ji} is a word in document $\vec{d_i}$. The documents are classified to the class that maximizes the probability $P(c_k|\vec{d_i})$, i.e. each document belongs to the class $argmax_{c_k}(P(c_k|\vec{d_i}))$.

B. Vector Space Model

The vector space model or term vector model [2][3] was one of the first methods to appear in the field of information retrieval. It is an algebraic method that represents documents as a set of vectors in the n-dimensional space, where n is the number of separate terms. The elements of these vectors, also known as term weights, can be calculated in many different ways, including the TF-IDF weighing (analyzed in Section III-B). The similarity between two documents $\vec{d_i}, \vec{q}$ is measured by the cosine similarity:

$$cos(\vec{d_i}, \vec{q}) = \frac{\vec{d_i} \cdot \vec{q}}{||\vec{d_i}||||\vec{q}||}. \tag{3}$$

A document q is categorized in the same class as the most similar document d_i, according to Eq. (3).

C. Latent Semantic Analysis

LSA, also referred to as latent semantic indexing (LSI) [4][5], uses the singular value decomposition (SVD) from linear algebra to reduce the size of the TF-IDF matrices (see Sections III-B, III-C). The name of this technique originates from the fact that after the reduction in dimensionality the documents are represented in the set of concepts they refer to. The category of a new document can be found according to the cosine similarity of Eq. (3).

D. Recurrent Neural Networks

A number of factors, such as the overcome on the vanishing gradient problem [6], the enormous collections of labeled data available today and advances in the use of GPUs and FPGAs, have resulted in an ever-increasing popularity of recurrent neural networks (RNNs) in many fields. Many tasks in computational linguistics, such as TC [7], dialogue generation [8], sentence simplification [9], summary extraction [10] are now mainly performed with the training of RNNs.

III. Text Categorization with GMMs

A. Training and Testing Data

The dataset that was used in this project came from recorded phone calls to the customer help center of a phone company. In this customer help center work 15 phone operators, each one specialized to a different issue. Each phone call is assigned to the most suitable phone operator. Our data consist of sentences that the customers uttered, converted into text, and the class label for each sentence. The label is a number indicating the phone operator who is an expert on each customer's issue. The task here was to create a classifier that could assign the phone calls to the most appropriate operators. For this project we assumed that the conversion from voice signal to text and the preprocessing of the data was correct. The number of training and test samples of each class are shown in Table I.

TABLE I
Data Sets

Class Number	Training Samples	Testing Samples
1	1141	122
2	1243	175
3	1211	129
4	260	48
5	601	62
6	38	6
7	185	20
8	477	71
9	46	6
10	25	7
11	112	13
12	113	11
13	148	18
14	112	21
15	807	124

B. TF-IDF representation of the data

In order to use GMMs the data has to be transformed into some arithmetical representation. For this we used the TF-IDF approach. We first created the dictionary, i.e. the set of all the different words that are used in the training data. We then computed the term-document matrix. Each row of the term-document matrix represents a document which, in our problem, corresponds to a sentence in the training data. The columns of this matrix represent the words of the dictionary. A cell of the term-document matrix holds the value of how many times a word has been encountered in a document.

The term-document matrix is then transformed into the TF matrix, that holds information indicative of the importance of a word in a document. For each document entry the number of times a word is encountered in this document is divided by the total number of words in the document. The term frequency of a term t in a document d is given by:

$$TF(d,t) = \frac{Number\ of\ times\ t\ appears\ in\ d}{Total\ number\ of\ terms\ in\ d}. \quad (4)$$

However, that information alone is not very useful for the classification task. This is due to the fact that connective words, pronouns etc. are very commonly used in all sorts of text but hold little information about the context. The large TF values that this type of words have in documents are not helpful in discriminating between different document contexts and do not facilitate the categorization task. For that reason the TF values are weighted by an IDF value that indicates the contextual importance of the word. An IDF matrix is created to hold for each word in the dictionary a value of the logarithmically scaled inverse fraction of the documents that contain the word. As shown in Eq. (5), the IDF values are obtained by dividing the total number of documents in the corpus by the number of documents d that contain the term t, and then taking the logarithm of that quotient:

$$IDF(t) = ln\left(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ containing\ t}\right). \quad (5)$$

The IDF value indicates how rare a word is. Very common words that are encountered in almost every test have an IDF value that approaches zero, while rare words have a large IDF value.

Very rare occurrences are also not indicative of the context of the document. The information of the TF and IDF matrices is combined in creating the TF-IDF matrix where the term frequencies are weighted by the inverse document frequencies. The TF-IDF transformation is widely used when working with text corpora but is also adopted in other applications as well like in recommendation systems [11][12], where the matrix corresponds to users and movies instead of documents and words. The TF-IDF scheme has also been proposed for citation ranking in [13].

C. Feature Reduction

The dictionary that we extracted from the training data had a size of about 3300 words. These words were used as the

features in our model. The dimensionality of the problem had to be significantly reduced to become feasible. To reduce the features of the problem we worked in two steps; first we downsized the size of the dictionary and the TF-IDF matrix by discarding the least important words manually and then performed SVD to downsize the feature space to the final number of features that were used.

For the first step of the feature reduction process we sort the terms in the dictionary and the TF-IDF matrix according to the importance of the words. For this we relied on the information contained in both the TF and IDF matrices. A mean TF value for each term was calculated and weighted by the IDF value of the word. This gives an indication of the general importance of each word. The tables were sorted according to this importance and the least important terms were discarded. We also tried to sort according only to the IDF values. That approach however gave very poor accuracy results, as was expected, since the very rare words do not usually contain enough contextual information to make a valid generalization for the context of the whole document.

After that we performed SVD on the TF-IDF matrix using Python's sklearn module $truncatedsvd$. SVD, also known as LSA, is used to reduce the high-dimensional feature space into a lower-dimensional space. The term and document vectors can now be treated as a lower-dimensional "semantic space", whose dimensions do not relate to any comprehensible concepts.

D. GMM

With the features space having a reasonably small dimension we used GMMs to estimate the distributions of the data samples over the feature vector for each class in continuous state space. From the TF-IDF we extracted the training samples of each class separately and used these samples to train a GMM for each class, using Python's sklearn module GMM. We assumed diagonal covariance matrices since our data set was not large enough.

E. Classification

For the task of classification we transformed our training samples to the reduced feature space accordingly. We first calculated the TF, IDF and TF-IDF matrices for the test samples, based on the downsized dictionary and then transformed the TF-IDF matrix of the testing data to the feature space that resulted from the SVD.

To classify the test samples we used ML classification. We calculated the log-likelihood probabilities for each class, as shown in Eq. (6):

$$ln(P(test\ sample|sample\ came\ from\ GMM\ of\ class)). \quad (6)$$

In other words we calculated the log-likelihood probabilities of a given sample coming from a distribution of each class, represented by a GMM. The test sample was then assigned to the class that has the maximum log-likelihood probability.

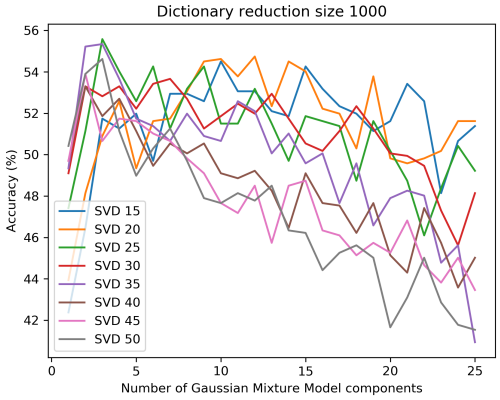

Fig. 1. Accuracy Results.

IV. Experiments

We ran experiments changing various parameters of the problem. Specifically we examined the accuracy of the classification task for the cases of

- initial dictionary size reduction of 0, 500, 1000, 1500, 2000, 2500, 3000 words
- final feature space (SVD components) of 15, 20, 25, 30, 35, 40, 45, 50
- number of Gaussian mixture components between [1, 25]

As mentioned previously the SVD/LSA dimensionality reduction results in a representation of semantic concepts. Hence we assumed that for the 15 different classes in our problem we would need at least 15 different semantic concepts in the feature space. The number of the Gaussian distributions used was limited by the case of the class with the minimum number of training samples in the data set. The accuracy was computed as the percentage:

$$\underset{percentage}{Accuracy} = \frac{Correctly\ classified\ samples}{Total\ number\ of\ testing\ samples}. \quad (7)$$

We also examined the accuracy for each class separately. The accuracy levels that were achieved range from 40 to almost 56%.

V. Conclusion

A. Results

A sample of the results is shown in Figure 1. These correspond to accuracy results that were achieved for an initial manual reduction of 1000 terms from the dictionary, for the various cases of feature sizes and GMM components. The overall maximum accuracy (55.58%) for all cases was achieved for an initial dictionary reduction of 1000 terms, 25 features and 3 GMM components.

One thing that became clear from examining the accuracy results for each class is that for the classes that have a very limited number of training and testing samples, such as classes 10, 6 or 9, the accuracy is close to zero, whereas for classes that have overwhelmingly more training samples, like 1 or

2, the accuracy is more than 80%. For classes with very few training samples the GMMs were not sufficiently trained, ergo their effectiveness is poor.

B. Future Work

In this project we examined the accuracy performance of the classifier while changing some parameters but there are other variations that could be tested as well. First and foremost there are quite a few versions of TF transforms that are said to improve the performance of categorization in some cases, e.g. binary transformation, logarithmic transformation, entropy of the terms, etc. Furthermore, classes are not equally represented in the training and testing data. A cross-validation method could probably yield better accuracy results.

Another factor that may have an effect in the accuracy of the categorization is the fact that, in language, words in a sentence are not independent utterances. The next word is always strongly associated with the context and the words previously used. Hence a very logical assumption would be that by considering pairs of words the results would be better. However, the connections between words are also relevant to the grammar and syntax of the language as well. The statistical information extracted from pairing words, especially when the training data set is small, is not always helpful. Nevertheless, an approach considering the mutual information, on top of the pairs of words $I(w_1, w_2) = log_2\left(\frac{P(w_2|w_1)}{P_(w_2)}\right)$ could possibly yield interesting results.

Another popular approach to TC in the recent years has been the vector method as mentioned in Section II. For approaches using similarity metrics we propose the use of the edit distance. As mentioned in [14] the edit distance is commonly used in cases of sentences that have varying lengths, as was our data set. For example a representative sentence could be chosen for each class or formed by combining elements from the training sentences. Then a test sentence would be categorized to the class that minimizes the edit distance between test and representative sentence.

Recent advances in deep learning are having a strong impact across a broad range of fields and in many (if not all) disciplines. Particularly relevant with the scope of NLP and TC, RNNs are now employed in almost every task in computational linguistics. The outstanding performance of RNNs will revolutionize the field of NLP and introduce a novel view in the subject of language, even on a more fundamental level by potentially revealing hints at linguistic structures and their development.

ACKNOWLEDGMENT

The authors would like to thank V. Diakoloukas, V. Tsiaras and V. Digalakis for their help and advice during this project.

REFERENCES

[1] A. McCallum and K. Nigam, "A Comparison of Event Models for Naive Bayes Text Classification," in *Technical Report WS-98-05*. The AAAI Press, 1998.

[2] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, Jan. 1988. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0306457388900210

[3] G. Salton and M. E. Lesk, "Computer Evaluation of Indexing and Text Processing," *J. ACM*, vol. 15, no. 1, pp. 8–36, Jan. 1968. [Online]. Available: http://doi.acm.org/10.1145/321439.321441

[4] G. W. Furnas, S. Deerwester, S. T. Dumais, T. K. Landauer, R. A. Harshman, L. A. Streeter, and K. E. Lochbaum, "Information Retrieval Using a Singular Value Decomposition Model of Latent Semantic Structure," in *Proceedings of the 11th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '88. New York, NY, USA: ACM, 1988, pp. 465–480. [Online]. Available: http://doi.acm.org/10.1145/62437.62487

[5] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[6] R. Pascanu, T. Mikolov, and Y. Bengio, "On the Difficulty of Training Recurrent Neural Networks," in *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ser. ICML'13. Atlanta, GA, USA: JMLR.org, 2013, pp. III–1310–III–1318. [Online]. Available: http://dl.acm.org/citation.cfm?id=3042817.3043083

[7] S. H. Lee, D. Levin, P. Finley, and C. M. Heilig, "Chief complaint classification with recurrent neural networks," *arXiv:1805.07574 [cs, stat]*, May 2018, arXiv: 1805.07574. [Online]. Available: http://arxiv.org/abs/1805.07574

[8] J. Li, W. Monroe, A. Ritter, D. Jurafsky, M. Galley, and J. Gao, "Deep Reinforcement Learning for Dialogue Generation," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 1192–1202. [Online]. Available: https://aclweb.org/anthology/D16-1127

[9] X. Zhang and M. Lapata, "Sentence Simplification with Deep Reinforcement Learning," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, pp. 584–594. [Online]. Available: https://www.aclweb.org/anthology/D17-1062

[10] S. Narayan, S. B. Cohen, and M. Lapata, "Ranking Sentences for Extractive Summarization with Reinforcement Learning," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 1747–1759. [Online]. Available: http://www.aclweb.org/anthology/N18-1158

[11] N. Good, J. B. Schafer, J. A. Konstan, A. Borchers, B. Sarwar, J. Herlocker, and J. Riedl, "Combining Collaborative Filtering with Personal Agents for Better Recommendations," in *Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence*, ser. AAAI '99/IAAI '99. Menlo Park, CA, USA: American Association for Artificial Intelligence, 1999, pp. 439–446. [Online]. Available: http://dl.acm.org/citation.cfm?id=315149.315352

[12] M. J. Pazzani and D. Billsus, "Content-Based Recommendation Systems," in *The Adaptive Web*, ser. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2007, pp. 325–341.

[13] J. Beel, C. Breitinger, and S. Langer, "Evaluating the CC-IDF citation-weighting scheme: How effectively can Inverse Document Frequency (IDF) be applied to references?" in *Proceedings of the 12th iConference*, 2017, p. 12.

[14] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Fourth Edition*, 4th ed. Orlando, FL, USA: Academic Press, Inc., 2008.