Augmenting a Classifier Ensemble with Automatically generated class level patterns for Higher Accuracy

Arthi Venkataraman
Wipro Technologies
Bangalore, India
Arthi.venkat@wipro.com

***Abstract*— Different types of classifiers were investigated in the context of classification of problem tickets in the Enterprise domain. There were still challenges in building an accurate classifier post data cleaning and other accuracy improving pre-processing techniques. Creating an ensemble of classifiers gave better accuracy than individual classifiers. The maximum accuracy was got by enhancing the ensemble with an additional automatically generated domain specific class wise keyword list. Use of this system gave us greater than 4 percent improvement over the techniques of just using the ensemble classifier. A further improvement in accuracy was obtained when a semi-supervised approach was followed where the automatically generated class level keys are further reviewed by domain team before usage.**

Keywords*— Natural Language Processing; Machine Learning; Classification; Committee-based approach; Feature extraction; ensemble; F-score; Accuracy; Latent Semantic Analysis; Term Frequency Inverse Document Frequency*

I. INTRODUCTION

Employees face a lot of issues across different functions in the organization like Payroll, Infrastructure, Facilities, Applications, etc. When employees raise a ticket they are traditionally asked to manually select a category for their ticket from a hierarchical menu driven interface. This leads to a lot of wrong selection of choice by end user. This in turn causes tickets to be raised in wrong bucket and delays the resolutions due to re-assignments.

We have built a system which accepts a Natural language text input from end user and automatically classifies the ticket in the right category. Key challenges in building such a system are the inaccurate training corpus, many closely separated classes, imbalanced classes, many classes with too less data and a need to handle a large number of requests. The challenges faced and technical steps needed to build such a system are described in detail in Arthi [2015] by same author. The focus of this paper is on the techniques to further improve the accuracy of the classification. The described system is in live use. This is an industry applied research paper.

II. BACKGROUND

A classifier system was designed which understands the natural language description entered by end user to automatically log the incident in the correct category. While this significantly improved the classification accuracy there were still issues with accuracy of this system. There was a problem specifically with classes with very low ticket volumes.

As a premise for this paper we assume the necessary data cleaning and other accuracy improvement techniques as described in Arthi [2015] are already applied. We focus on the actual classification algorithms and how the accuracy of prediction using same can be improved using the approach we have specified in this paper.

III. REFERENCES

There is a lot of research addressing the techniques to build more accurate classification systems. Few of the key ones are highlighted in this section.

In Pu Wang [2007] it has been found that the classification results with encyclopedia knowledge are much better than the baseline "bag of words" methods. Evigeniy [2006] has discovered that use of a wider body of knowledge in the classification process enables a more intelligent and accurate classification.

In Ricardo [2003] classifiers are augmented by creating random subsets of the features and building classifiers for these subset of features. The resulting classifiers are then ensembled for getting more accurate classifications.

In Ariel [2009] it is shown that we can have large accuracy gains using automatically extracted training data at much lower cost.

In Chade [2004] large accuracy in training is obtained with the use of automatically extracted bi-grams as part of the classification process.

In Chuang [2010] novel techniques like shuffling datasets to create different decision boundaries in several ways, random subspace, and bootstrapping has been attempted. The smoothing-average method is found to result in higher classification accuracy.

In Andreas [2008] the relationship between accuracy and number of features is evaluated.

In Smith [2011] it is shown that by identifying and removing instances which are misclassified the classifier accuracy increases.

978-1-4673-9606-6/15/$31.00 ©2015 IEEE

In Andrew [1999] a technique is discussed where a classifier is built just by using a set of keywords per class.

In Nigam [2000] it is shown that the accuracy of learned text classifiers can be improved by augmenting a small number of labeled training documents with a large pool of unlabeled documents. An improvement of thirty percent is seen using this technique of combining expectation maximization and naïve Bayes classifier.

In Zaiane [2002] the use of association rule mining in building a text categorization system is explored. The approach is found to have fast training phase, and the rules of the classifier generated are easy to understand and manually tunable.

In Matsuo [2009] it is shown that using both Pair wise and similarity based clustering enhance the quality of the automated extraction of top key terms.

In Feifan [2009] it is shown that simple unsupervised TFIDF approach performs reasonably well and the additional information from POS and sentence score helps keyword extraction

In Yanhong [2014] used word2vec for describing a document along. The resulting word2vec vector was clustered and a new label assigned to each cluster. Use of SVM based classification for these new labels gave good accuracy.

In Lawrence [2008] Deep Belief Nets are used for clustering large document collections. This is followed by building a classifier with the different cluster labels as class labels.

In Yanhong [2014] used word2vec for describing a document along. The resulting word2vec vector was clustered and a new label assigned to each cluster. Use of SVM based classification for these new labels gave good accuracy. This is similar to the approach followed in Lawrence [2008], the only difference being the use of word2vec for the clustering step.

In Arthi [2015] techniques and solution for building a large scale accurate classifier is discussed.

IV. Overall Solution Approach

Figure 1: Overall Solution Diagram

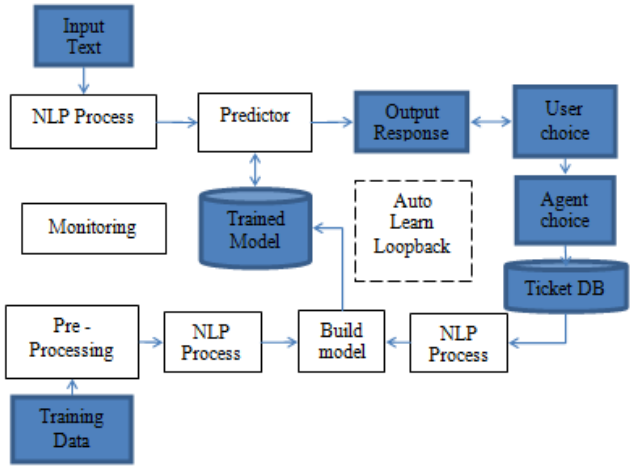


Figure 1 represents the main high level modules of the overall solution: Components of this solution are described in detail in Arthi [2015] (paper by author)

978-1-4673-9606-6/15/$31.00 ©2015 IEEE

Figure 2: RELEVANT BLOCKS for Current Paper

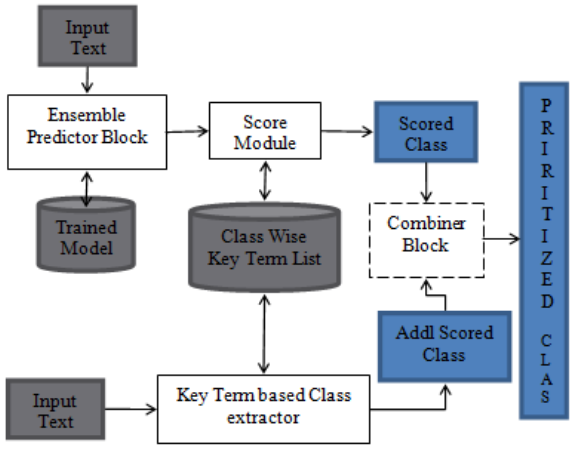


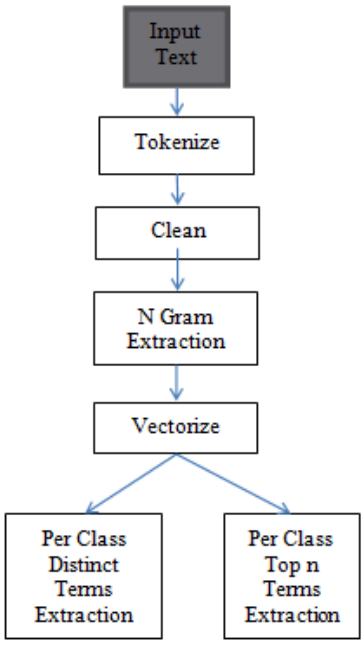
Figure 2 highlights the key functional blocks which are relevant for this paper. The key functional blocks are described below. The blocks are split under the headers Training and Prediction. Blocks which are active during training are put under training. The blocks which are active during prediction are put under prediction.

A. Training Stage

1) Building the Domain Specific Key Terms

We are interested in extracting the most distinguishing terms from the unstructured text for a class of tickets. This needs to be done in an automated manner. Figure 3 shows the key steps in this process. We have used a combination of statistical and linguistic based approach.

Figure 3: Automated Key Term Extraction



Solution will read the training data. A class wise split of the training data will be automatically done. The text will be split into tokens (individual words). The cleaning block will remove:

- White space / Special Characters
- Stop words
- Numbers
- Any other un-wanted name entities

Post cleaning it will be passed through an n-gram extraction routine. Key steps in this are:

- Get the expected n-gram range
- Iterate through the input text
- For every word in input text extract the different n-gram combinations

The processed n-grams will be passed through the vectorization process. The vector will be a matrix where

- Every term (n-gram) will have one column.
- Every class will have a row
- At the cross of a class and term , the number of occurrences of that term in the class will be represented

TF-IDF (Term Frequency Inverse Document) normalization will be done for this vector.

Post this the top terms (based on frequency of occurrence) for every class will be retrieved

The top distinct terms for a class will then be retrieved. This is done using the Chi-Squared test and selecting only the top n terms post this step.

The generated terms are then passed through a Statistical Substring Reduction (SSR) algorithm. This is used to remove redundant N-grams from an N-gram set using N-gram statistics information. For example, if both "United States of America" and "United states" occur 20 times in the corpus, the latter will be removed.

We added a part of speech tagger. This enables us to retain only the key POS tagged components. We retrieved plural nouns and adjectives. The quality of the extracted keywords was good. However there was a challenge of domain specific terms also being removed when we used this technique.

The other technique we tried was to remove only certain named entities by using a named entity tagging approach. We removed terms which were proper nouns. We also identified irrelevant information including IP addresses, email ids, time and date using regular expressions. These were then removed before applying the keyword extraction algorithm. Using this technique enabled us to get an accurate key term extraction and the domain specific terms could also be retrieved using this.

We are working on further enhancing the automated key term extraction block. We will calculate the mutual information

978-1-4673-9606-6/15/$31.00 ©2015 IEEE

scores of terms. We will retain only those terms per class which will have maximum impact on the accuracy of the classification. We plan to further augment the results using a machine learning approach. We will build a combiner logic which will rank and combine the terms returned by the different techniques. We will retain the top n (configurable number) ranked relevant terms post this.

2) Supervised Cleansing of Per class Key Terms

The terms extracted from previous step will be shared to the domain team for a review. They can remove, add or modify the terms to improve the key term list. This is an optional step

3) Building Classifiers

a) Choice of Classifiers

Different algorithms are available as part of the Model tool box. Chosen classifier algorithms will have following characteristics:

- Is a supervised machine learning algorithm
- Handles text classification
- Can handle the size of the training set
- Prediction should be in near real time

A one-time configuration of the set of chosen algorithms for the class of input data is done.

b) Approach for shortlisting the algorithm:

An iterative process is followed to shortlist the correct algorithms. Different algorithms perform well with different kinds of data. Once the algorithms are chosen the different parameters for the algorithms is also tuned. Based on benchmarking of different algorithms the best performing algorithms are selected

- Parameters for selection will vary based on the use case
- Key Parameters which we considered in order of importance are :
    - F- Score, Precision, Re-call
        - Without good benchmarks in this system would give wrong classifications
    - Model Prediction Time
        - Near Real time system needed this
    - Model building Time
        - Certain algorithms did not complete the training cycle and hence rejected.

After every cycle the above parameters are verified. If there is a scope for improvement in these parameters then another cycle is performed till we get the required level of performance. If for any algorithm we are unable to reach the desired level of performance and there is no scope for improvement then the algorithm is rejected.

4) Trained model

This is the model store. The trained models are stored inside this store. The models in the store are updated whenever the build model block is triggered with new inputs.

B. Prediction Stage

1) Ensemble Predictor Block

In live use whatever incidents are entered by user are passed through NLP Process block. The output of this is then fed to the predictor. The predictor predicts the output classes using the models in the trained model store.

The different classes returned are combined. The models built by the shortlisted algorithms and stored in the model store are considered members of a committee. Each of the models predicts possible classes for the given input. It is possible for a model to return more than one class also. The different unique responses are then combined and given as an output of the Ensemble predictor block.

2) Scoring Module.

For each of the returned class of the Ensemble predictor block is passed to the Scoring module along with the sentence entered by the user. The scoring block extracts the key terms from the input sentence. It then does a comparison with keywords for that class in Keyword per Class list. It returns a score between 1 and 0 with 1 for complete match and a score of zero for no match. All classes returned from the ensemble predictor block are augmented with this score. The scoring block is described in detail in Section V.

3) Keyword Based Class Extraction

This module will take an unstructured sentence as input. It will extract top key terms from the sentence. It has a Semantic comparator logic which will compare the extracted terms to the domain specific terms of the different classes. (This was extracted in previous two steps.) It will return those classes which have a match percentage which are above the defined threshold. The selected classes with have associated with the match score. The technique of top pattern extraction followed is described in detailed in the Top Pattern Extraction technique section.

4) Combiner Block

This is the critical block which combines the classes returned from different modules and returns an ordered list of predicted classes. The combiner block can be configured to return an exact number of classes. The combiner block will work on following logic:

1. A class which appears as output across both Ensemble predictor block as well as keyword based class extractor block is put in first bucket.
2. A class which appears as output of only the Ensemble predictor block is put in second bucket.
3. A class which appears as output of only the keyword based class extractor block is put in third bucket.

Within a bucket the ordering is based on the score with the highest score occupying the first position.

978-1-4673-9606-6/15/$31.00 ©2015 IEEE

V. SCORING SIMILARITY MODEL

This model will accept two textual inputs. It will do comparison of the two inputs and return a score between zero and one. A score of zero means no match. A score of one means a complete match. There are alternate techniques to approach this

- Simple Distance based scoring.
    - Vectorize the two inputs. Post these find the Cosine Distance between the two vectors. Can replace cosine distance with any other relevant distance metric.
- Semantic Similarity between two inputs include
    - Word Similarity augmented using WordNet
    - Use LSA (Latent Semantic Analysis) as a measure of similarity between two inputs

VI. EXPERIMENTAL SETUP

A. Data set

For testing our algorithms we used tickets falling under different functional categories. This includes infrastructure, facilities, applications, payroll, travel, etc.

Below are the details of the overall data corpus:

Total classes – 2759

Data size (Number of training rows) - 495,000

TABLE 1: DATA COMPOSITION OF COMPLETE TRAINING SET

Number of Tickets in Bucket	*Number of Classes*
1.00	400.00
2.00	300.00
3.00	100.00
4.00	75.00
5.00	75.00
6 to 10	300.00
11 to 50	800.00
51 - 200	480.00
201 - 400	98.00
Greater than 400	131.00
Total Classes	2759.00

Below are the details for all tickets falling under the application management function.

Initial Data Analysis (obtained after basic clean up) showed the following attributes:

Total classes – 518

Data size (Number of training rows) - 72286

View of the Data spread across classes

TABLE 2: DATA COMPOSITION OF APPLICATION TRAINING SET

Number of Tickets in Bucket	*Number of Classes*
1.00	74.00
2.00	52.00
3.00	24.00
4.00	40.00
5.00	6.00
6 to 10	78.00
11 to 50	115.00
51 - 200	71.00
201 - 400	26.00
Greater than 400	32.00
Total Classes	518.00

B. Experimental Configurations

Our approach to this experiment is to use the different configurations for prediction. Each of the data set would be predicted using the different identified configurations the output will be measured for each of the configurations using the identified metrics. A comparison will then be arrived at for the performance across different configurations.

1) Configuration 1 – Single Classifier

In this case training and prediction is performed using a single algorithm which is the Ridge classifier.

2) Configuration 2 – Ensemble of Classifiers

In this case multiple algorithms were selected. The algorithms were Ridge, SVM and a couple of other in house proprietary classifiers.

The output of all these classifiers were combined using a Committee based voting approach. Classes which were predicted from multiple classifiers were given the highest weight.

3) Configuration 3 – Ensemble of Classifiers enhanced with per class automatically generated key terms

The same setup as used in Ensemble of classifiers is used here. In addition all the training tickets are passed through a unique pattern extraction tool developed for this. The output of this is a file with the list of top unique patterns associated with every class. This data along with the based training data is passed through the augmented classifier as described in the solution approach.

4) Configuration 4 – Ensemble of Classifiers enhanced with per class sem-supervised generated key terms

This setup is exactly same as Configuration 3. The key difference is that the list of terms generated per class is now reviewed and augmented by a domain team.

978-1-4673-9606-6/15/$31.00 ©2015 IEEE

VII. Evaluation Metrics

We selected the F-score (Harmonic mean of precision and recall) as the overall evaluation metrics. We attempted to get the score at

- Overall system level
- Individual Class levels

VIII. Results

Each of the data sets was sent through the different configurations.

Figure 4 shows the result of predicting using the different configurations for the application group.

TABLE 3: F-SCORE ACCURACY ACROSS CONFIGURATIONS

Function	*Single Classifier F-score*	*Ensemble Classifier F-score*	*Un-Supervised Enhanced Ensemble Classifier F-score*	*Semi-Supervised Enhanced Ensemble Classifier F-score*
Application	71%	78%	82%	85%

From figure 4 we can see following:

- Using an ensemble of classifiers gives a better result than using just a single classifier
- Using our augmented classifier technique gives better results than just the ensemble
- Using the augmented classifier in a semi supervised mode gives us a better performance than using the classifier in an un-supervised mode.

Figure 4: F-score accuracy across configurations

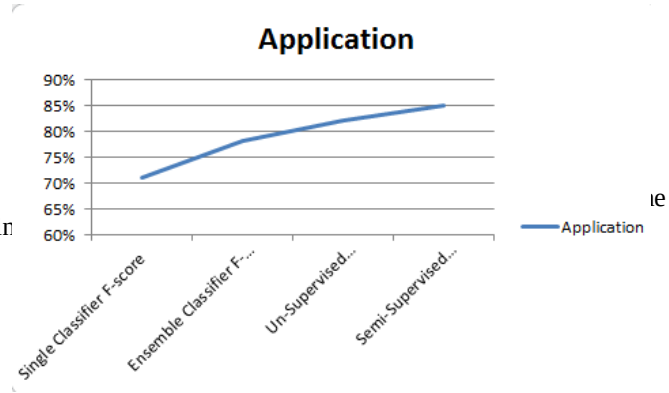


TABLE 4: F-SCORE ACCURACY ACROSS CONFIGURATIONS

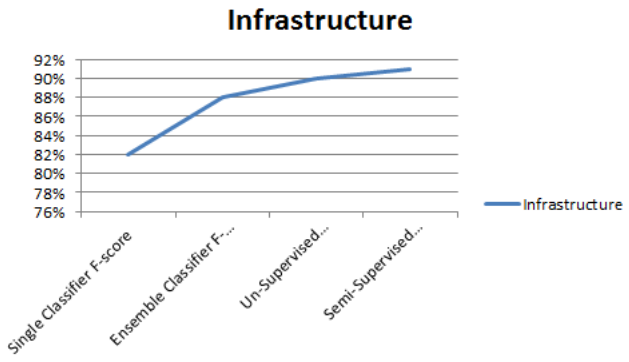
Function	*Single Classifier F-score*	*Ensemble Classifier F-score*	*Un-Supervised Enhanced Ensemble Classifier F-score*	*Semi-Supervised Enhanced Ensemble Classifier F-score*
Infrastructure	82%	88%	90%	91%

Similar conclusions can be drawn from these results.

We see the relative improvement in set two is less compared to the set 1. Our inference is that we can attribute this to the following:

- The base accuracy is much higher is set two
- Also the class level spread is much less in set two

Figure 5: F-score accuracy across configurations



We see that the proposed augmented classifier approach works well when there are many classes with very less tickets. In the normal classifier approach these classes are overwhelmed by the classes with more number of tickets.

For example if the issue is Eclipse installation, the normal classifier places this in Desktop installation. However the augmented classifier rightly identifies this class as Eclipse Application type of ticket.

IX. CONCLUSION

It is a daunting task to create a working prediction system with good level of accuracy and performance for a large scale live system while supporting diverse types of classes. Text classification in real life is a challenging issue. Supervised classification approach will need to be augmented using ensemble techniques, scoring models as well as additional models based on per class level keywords. The augmented classifier model described in this paper gave good results especially in case where classes had very less training tickets. We have seen a minimum of four percent improvement in classification accuracy. Using the augmented classifier in a semi supervised mode performed better than using same in a completely un-supervised manner.

978-1-4673-9606-6/15/$31.00 ©2015 IEEE

X. FUTURE WORK

We plan to explore alternate classification techniques. . Using deep belief networks for classification is an approach we would like to explore. We will investigate more accurate classification using solutions like word2vec. Use of word2vec to cluster the documents followed by classification will help resolve the issue of less separated classes. Classes which are less separate would be merged during clustering. Hence classification on these new labels would give higher accuracy.

We need to explore alternate techniques to extract the domain specific keyword list more accurately. One approach could include using a few key seed terms for class and use same to pick .additional terms from the tickets for the class using word to word similarity approach. We can further augment the terms created per class using online resources such as WordNet.

ACKNOWLEDGMENT

I give my heartfelt thanks to my organization Wipro who has enabled me to build this system. They have also provided the needed support to make this presentation. I would like to specially thank Mr. Ramprasad.K.R, Senior Fellow, Distinguished Member of Technical Staff, Wipro who has been my mentor and helped with the review of this paper. I would like to thank my family and specially my spouse and parents for all their support.

REFERENCES

[1] "Improving Text Classification by Using Encyclopedia Knowledge by Pu Wang ; Peking Univ., Beijing ; Jian Hu ; Hua-Jun Zeng ; Lijun Chen. Data Mining, 2007 ICDM 2007. Seventh IEEE International Conference on Data Mining, Pages 332 -341

[2] Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge by Evgeniy Gabrilovich and Shaul Markovitch Department of Computer Science Technion—Israel Institute of Technology, 32000 Haifa, Israel {gabr, shaulm}@cs.technion.ac.il (Unpublished)

[3] Attribute bagging: improving accuracy of classifier ensembles by using random feature subsets by Robert Bryll, Ricardo Gutierrez Osuna, Francis Quek in Pattern Recognition Volume 36, Issue 6, June 2003, Pages 1291–1302

[4] Improving Classification Accuracy Using Automatically Extracted Training Data by Ariel Fuxman, Anitha Kannan and others. Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining Pages 1145-1154

[5] The use of bigrams to enhance text categorization by Chade-Meng Tan,Yuan-Fang Wanga,,Chan-Do Lee Information processing & management, 2002

[6] Novel techniques to increase classification accuracy in machine learning by Tzu-Cheng Chuang, Purdue University (Unpublished)

[7] On the Relationship between Feature Selection and Classification Accuracy Andreas G. K. Janecek andreas.janecek@univie.ac.at Wilfried N. Gansterer wilfried.gansterer@univie.ac.at, FSDM, 2008

[8] Improving Classification Accuracy by Identifying and Removing Instances that Should Be Misclassified Michael R. Smith and Tony Martinez. IJCNN, the 2011 International Joint Conference on Neural Networks, Pages 2690 - 2697

[9] Text Classification by Bootstrapping with Keywords, EM and Shrinkage Andrew McCallum mccallum@justresearch.com S Just Research 4616 Henry Street Pittsburgh, PA 15213 Kamal Nigam t knigam@cs.cmu.edu School of Computer Science Carnegie Mellon University Pittsburgh, PA 15213 (Unpublished)

[10] Text Classification from Labeled and Unlabeled Documents using EM KAMAL NIGAM knigam@cs.cmu.edu School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA, ANDREW KACHITES Sebastian Thurn and others. Machine learning, 2000 – 39, 103–134, Springer

[11] Classifying text documents by associating terms with text categories By Osmar R Zaïane, University of Alberta. Journal Australian Computer Science Communications archive, Volume 24 Issue 2, January-February 2002 Pages 215 - 222 , IEEE Computer Society Press Los Alamitos, CA, USA

[12] Keyword extraction from a single document using word co-occurrence statistical information. By Y. Matsuo and M. Ishizuka, International Journal on Artificial Intelligence Tools, 13:2004, 2004.

[13] Unsupervised approaches for automatic keyword extraction using meeting transcripts. By Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. In Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, NAACL '09, pages 620–628, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[14] Document Classification using Deep Belief Nets by Lawrence McAfee, Unpublished

[15] A New Study Based on Word2vec and Cluster for Document Categorization by Yanhong YUAN, Liming HE, Li PENG, Zhixing HUANG. Journal of Computational Information Systems 10: 21 (2014) 9301–9308

[16] Machine learning techniques for building a large scale production ready classifier by Arthi Venkataraman. Proceedings of the Second International Conference on Data Mining, Internet Computing, and Big Data, Reduit, Mauritius 2015

978-1-4673-9606-6/15/$31.00 ©2015 IEEE