A literature review of approaches to the problem, containing at least 9 publications or other substantial references

This report presents our analysis of various approaches adopted in Text classification and categorization based on the study of various publications and research done by the team. We summarize our analysis of learnings and attempt to outline our approach towards doing Text Classification on Tweets.

Introduction

Text categorization (TC—a.k.a. text classification, or topic spotting), the activity of labeling natural language texts with thematic categories from a predefined set [Seb, 2002]. The amount of data generated in recent times has become enormous and hence there is an urgent needs more than ever before to classify text and make sense of this vast information.Text classification find its application various domains from email classifications to automated document indexing, automated metadata formation , cataloging of web resources.

Popular Approaches

There have been many popular approaches to Text classification, in earlier times a lot of text classification was done using an operational technique of *Knowledge engineering* [3] which involves defining rules on how to classify documents under a given category. This was a tedious activity and often required investment of large amount of time, also with the increase in the formats of text it became increasingly difficult to redefine and create new rules.

A more popular technique of Text classification which has become increasingly popular in recent times is based on *Machine learning*.This approach requires building an automatic text classifier by learning the characteristics of the categories from training set. This saves a lot of time and skilled manpower and the Classification accuracy is better than that of classifiers built by knowledge engineering methods.

Often the term Text classification means different things to different people (1) assigning of documents to a set of predefined categories based on the text (2) automatically identifying new categories and grouping the the documents under them , this is also called Text clustering [Merkl,1998]. In our project we will attempt to do the first part i.e. assigning documents (in our case tweets) to a set of predefined categories using Supervised Machine learning techniques.

Formal definition of Text Classification

Given a set of previously unseen documents D = $\{d_1, d_2, d_3 \ldots.\}$ and a set of predefined classes or categories C = $\{c_1, c_2, \ldots. c_k\}$, a classifier (categorizer) is a function **κ** that maps a document from set D to the set of all subsets of C. A value of *T* assigned to (d_j, c_i) indicates a decision to file d_j under c_i, while a value of *F* indicates a decision not to file d_j under c_i. Function k also called the classifier model is defined as k: D X C $\rightarrow$ {*T*, *F*}

Categories are just labels. In our case we use categories that are primarily non overlapping.

Types of Text Classification

Text classification can be of various types single label vs multiple label and Hard categorization vs. Ranking categorization.

Single Label vs Multiple Label approach: The case in which exactly one category can be assigned to a document in called single label categorization. A popular approach for single label categorization is binary text classification when the document must be assigned either to category ci or its complement. In multiple label approach a document can be classified in multiple categories. In our project we are using the multiple label approach in which each document (tweets) can be classified under multiple categories.

Hard categorization vs. Ranking categorization: In hard categorization is fully automated and classifier decides on document's category, whereas in ranking categorization a partially automated systems and generate ranked categories.

Process of Text Classification

Training Set, Test Set, and Validation Set

The ML approach relies on the availability of an initial corpus $\Omega = \{d_1, d_2, \ldots, d_{|\Omega|}\} \subset \mathcal{D}$ of documents pre-classified under the category $C = \{c_1, c_2, \ldots, c_{|C|}\}$. A document d_j is a positive example of category c_i if $\Phi(d_j, c_i) = T$, and a negative example of c_i if $\Phi(d_j, c_i) = F$. In research settings (and in most operational settings too), once a classifier Φ has been built it is desirable to evaluate its effectiveness. In this case, prior to classifier construction the initial corpus is split in two sets, not necessarily of equal size:

1. A training (and validation) set $TV = \{d_1, d_2, \ldots, d_{|TV|}\}$. The classifier Φ for categories $C = \{c_1, c_2, \ldots, c_{|C|}\}$ is inductively built by observing the characteristics of the documents;
2. A test set $T_e = \{d_{|TV|+1}, \ldots, d_{|\Omega|}\}$, used for testing the effectiveness of the classifiers. Each d_j is fed to the classifier, and the classifier decisions of the training set classifier are compared with the expert decisions . A measure of classification effectiveness is based on how often the values of classifier built with training set match with classifier for test set.

The documents in the test set cannot participate in any way in the inductive construction of the classifiers; if this condition were not satisfied, the experimental results obtained would likely be unrealistically good, and the evaluation would thus have no scientific character [Mitchell1996, page 129].

Cleaning of data

It is very important that the data used is in a format that is consistent(to a certain extent and clean). To get best results from the classifier following methods (at least) should be adopted.

Stop word removal: In computing, *stop words* are words which are filtered out prior to, or after, processing of natural language data (text). There is not one definite list of stop words which all

tools use, if even used. Some tools specifically avoid removing them to support phrase search. Any group of words can be chosen as the stop words for a given purpose. For some search machines, these are some of the most common, short function words, such as *the*, *is*, *at*, *which* and *on*.In our case we filter ut commonly used words like are,is, where, who.

Stemming: This is grouping words that share the same morphological root), its suitability to TC is controversial. Although, similarly to unsupervised term clustering of which it is an instance, stemming has sometimes been reported to hurt effectiveness (e.g., Baker and McCallum [1998]), the recent tendency is to adopt it, as it reduces both the dimensionality of the term space and the stochastic dependence between terms

Document Indexing

Texts cannot be directly interpreted by a classifier or by a classifier-building algorithm. Because of this, an indexing procedure that maps a text d_j into a compact representation of its content needs to be uniformly applied to training, validation, and test documents. The choice of a representation for text depends on what one regards as the meaningful units of text (the problem of lexical semantics) and the meaningful natural language rules for the combination of these units (the problem of compositional semantics). A text d_j is represented as a vector of term weights $d_j = \langle w_{1j}, w_{2j}, ..., w_{|T|j} \rangle$ where T is the set of terms (sometimes called features) that occur at least once in at least one document of Tr, and $0 \leq w_{kj} \geq 1$ represents, loosely speaking, how much term t_k contributes to the semantics of document d_j.

In order to create feature vectors we can adopt various approaches.e.g. a ***binary weight*** can be used where 1 denotes the presence of a term in the document and 0 denotes the absence of term. In the non binary approach people often use ***tf-idf*** [Salton and Buckley,1988] defined as:

$$tfidf(t_k, d_j) = \#(t_k, d_j) \cdot \log \frac{|Tr|}{\#_{Tr}(t_k)}$$

where $\#(t_k, d_j)$ denotes the number of times t_k occurs in d_j, and $\#Tr(t_k)$ denotes the document frequency of term t_k, that is, the number of documents in Tr in which t_k occurs. This function embodies the intuitions that (i) the more often a term occurs in a document, the more it is representative of its content, and (ii) the more documents a term occurs in, the less discriminating it is. EXPERTS [Cohen and Singer 1999]). In order for the weights to fall in the [0,1] interval and for the documents to be represented by vectors of equal length, the weights resulting from tfidf are often normalized by cosine normalization , given by:

$$w_{kj} = \frac{tfidf(t_k, d_j)}{\sqrt{\sum_{s=1}^{|T|}(tfidf(t_s, d_j))^2}}$$

Classification Methods

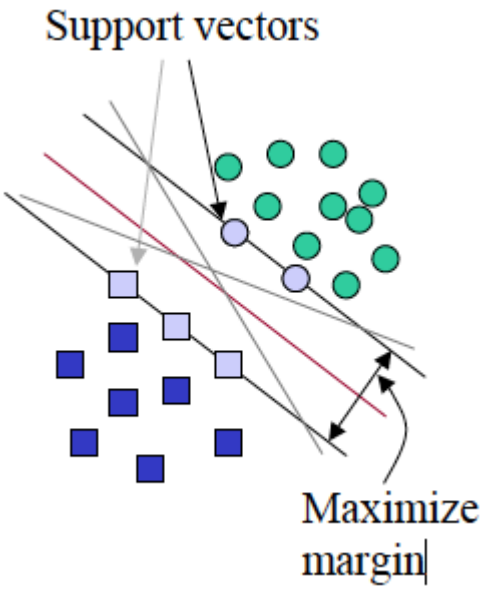
As described in the Approaches, the text classification system can be either supervised or unsupervised. In case of a supervised classifier, it has to be trained based on a set of predefined classification. Hence it needs external source of information in order to carry out classification. In unsupervised user does not have to specify any information about the features. In our project we use the supervised learning techniques. Some of the most well known techniques are:

Naive Bayes classifiers: Naive Bayes is a type of classifier that relies on a probabilistic model that assumes independence among the features. This means that this model assumes that the presence of a feature of one category is independent of the presence of a feature in a not(like spam/non spam for ex.), but might fail in cases where the data set is sparse / inconsistent, which is true in our case (tweets). Using Naive BAyes for tweets might not work very well mainly due to the structure of the tweet themselves.
The Naive Bayes classifier is closely related to the Binary Independence Model. In general, the Naive-Bayes classifier assumes that the features are not associated with each other (identical to the BIM). A feature can be dened as an attribute which helps to identify that object. the Naive Bayes classifier assumes that none of those features are related to each other. While this is a rather simplifying assumption (hence the name Naive), it turned out that this algorithm performs remarkably well in practice, even when strong feature dependencies are present [Pedro,Pisani, 1997].

KNN: The `k Nearest Neighbor' approach assigns a class to each training document. Then it classifies a new document by assigning it to the same class as the k nearest neighbors. For example, if k is set to 1, the new document will have the same class as the immediate neighbor. If k is 5, the algorithm will choose the class which occurs most often in the surrounding 5 neighbors. kNN (with k > 1) performs quite well. On a Reuters-21578 data set, kNN performed better than NB (82.6% accuracy vs. 72.3%), but not as good as SVMs (86.4%) [Horn, Thesis,2010]

SVM: This is one of the most popular techniques used in Text Classification. The basic Basic idea of support vector machine is to create an optimal hyperplane for linearly separable patterns , it extend to patterns that are not linearly separable by transformations of original data to map into new space – Kernel function. SVM algorithms are commonly used for pattern recognition.
Support Vectors are the data points that lie closest to the decision surface. These are the points that are most difficult to classify. They have a direct bearing on the optimum location of the decision surface.

The data is divided into hyperplanes and SVM machines maximize the margin around the separating hyperplane. The decision function is fully specified by a subset of training samples, the support vectors.



Evaluation of Text Classifier

As for text search systems, the evaluation of document classifiers is typically conducted experimentally , rather than analytically. The reason is that, in order to evaluate a system analytically (e.g., proving that the system is correct and complete), we would need a formal specification of the problem that the system is trying to solve (e.g., with respect to what correctness and completeness are defined), and the central notion of TC (namely, that of membership of a document in a category) is, due to its subjective character, inherently non formalizable. The experimental evaluation of a classifier usually measures its effectiveness (rather than its efficiency), that is, its ability to take the right classification decisions. Measure of classification is often done by precision and recall.

REFERENCES:

Fabrizio Sebastiani. Machine learning in automated text categorization. ACM Comput. Surv., 34(1):1{47, 2002.}

Support Vector Machine Active Learning with Applications to Text Classification. Simon Tong, Daphne Koller. *Proceedings of the Seventeenth International Conference on Machine Learning. 2000*

MITCHELL, T.M. 1996. Machine Learning. McGraw Hill, New York, NY.

COHEN, W. W. AND SINGER, Y. 1999. Context sensitive learning methods for text categorization. ACM Trans. Inform. Syst. 17, 2, 141– 173.

SALTON, G. AND BUCKLEY, C. 1988. Term-weighting approaches in automatic text retrieval. Inform. Process. Man. 24, 5, 513–523. Also reprinted in Sparck Jones and Willett [1997], pp. 323–328

Pedro Domingos and Michael J. Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. Machine Learning, 29(2-3):103{130, 1997}

Owen Phelan, Kevin McCarthy, and Barry Smyth. Using twitter to recommend real-time topical news. In RecSys, pages 385{388. ACM, 2009.}

MERKL, D. 1998. Text classification with self organizing maps: Some lessons learned. Neurocomputing 21, 1/3, 61–77.

BAKER, L. D. AND MCCALLUM, A. K. 1998. Distributional clustering of words for text classification.

A Tutorial on Support Vector Machines for Pattern Recognition CHRISTOPHER J.C. BURGES

Analysis and Classification of twitter messages, Master's Thesis, Graz University of Technology, 2010