Classification of short product reviews using semisupervised learning

David Przybilla
dav.alejandro@gmail.com, davida@coli.uni-saarland.de

Term paper for Unsupervised, semisupervised learning Seminar
Department of Computational Linguistics and Phonetics
Universität des Saarlandes

September 4, 2012

Abstract

Microblogging has become a trend,thus many users are using such platforms to share their opinions.Mining these opinions can reveal valuable information to companies, such as what features to develop in their products or marketing strategies.However there are major drawbacks when mining tweets, first their length do not provide enough word occurrences so traditional bag-of-words methods have limitations, second they are extremely noisy and third annotated data is limited and costly to get.
In this paper we address the problem of classifying short product reviews by using semisupervised learning. A comparison among two semisupervised methods and a supervised method is given. Finally we propose different ways to improve the given results by injecting knowledge into the semisupervised models.

1 Introduction

Twitter and Tuenti are popular microblogging services in Latinamerica, as prices of smartphones became more accessible to people big communities have grown around this services.
These communities are actively commenting everything from political events to products experiences. Therefore it has become of interest for many companies to analyze the microblogging data in order to understand what their clients want of their products, this led naturally to sentiment Analysis.

One of the subtasks of Sentiment Analysis is to find the topic or aspect of an opinion, for example given a set of opinions about a mobile phone, it is possible to label each comment with the topic that it is referring to, this labels could be “battery”, “design”, “operative system” etc. This Subtask can be modelled as a classification problem, given a set of fixed possible topics.

In the scenario of text classification semisupervised techniques present two advantages:
First a semisupervised method should allow to classify text by annotating only a small portion of data,thus reducing the labor of annotating.
Second the unlabelled data is used during training, this is important since the amount of unlabelled data in these tasks is abundant and available, therefore it can be used to provide extra knowledge to the model.

Zhang and Kubota [AZ05a] proposed a semisupervised approach using linear classifiers for multiple learning tasks.
They proposed to create additional classifications tasks (Auxiliary Problems) aside from the Target Problem.
The underlying idea is that the auxiliary problems will help finding good predictive structures.
One of the constraints in this approach is that

auxiliary problems should be able to automatically generate labelled data from the original unlabelled data. This method has been used in tasks such as Text Chunking [AZ05b]

In contrast [Zhu05] proposes a graph approach called label propagation for doing semisupervised learning.
This approach maps the data to a graph representation, then labelled instances propagate their labels through the graph, allowing unlabelled data to adopt the label of those instances which are similar.
[SUSB] uses Label Propagation for doing Polarity Classification on tweets.

Even though classifying text has been a widely studied topic,the focus has been on long documents, whereas tweets are at most 144 characters long.
The new trends in these social networks have led to research about classification in short texts, the latter has shown that it arises new challenges, and former approaches are not effective.
One of the reasons is that user generated comments in the mentioned services tend to be extremely short, leading to sparce feature representations.

In respect to short text classification Xinghua and Hongge [FH10] proposes to do Feature Extension to deal with data sparcity, in their approach each comment is extended with extra words from an expansion vocabulary.
[GM06] proposes on the other hand to use encyclopedic knowledge from wikipedia for extending the short comments.
As opposed to the above ones, [Sun12] reduces the word space of the comments to keywords and use information retrieval with a voting scheme to find labels for short comments.

This paper describes the setup of an experiment for classifying short-text comments of product reviews extracted from twitter.
Two semisupervised techniques are used, Label Propagation [Zhu05] and a variation of Structural learning problem for multitasks [AZ05a].
The next sections describe briefly the preprocessing of the tweets,each of the proposed semisupervised methods and how they were adapted to the task.
Addionally the results are shown, where both methods are compared among themselves and to a supervised approach. Finally a short conclusion and future work are discussed.

The source code of the implementation is available at github[1],the datasets however are not public, since they are constrained by a permission license.

2 Datasets

The Datasets of this experiments were provided by Meridean.
Meridean[2] is a Colombian company which extracts tweets mentioning latinamerican companies or products.

The datasets are separated by product.
There are 3 datasets in this experiment:

- Sportswear dataset
- Mobile-Phone dataset
- Hygienic-product dataset

Each dataset is composed of tuples,each tuple contains : the source of the opinion, the opinion, the polarity and the topic.

Comment: #ProbandoXperiaArcS Gracias @TalkMex. Las fotos se ven tan nitidas.
Translation: #TestingXperiaArcs Thanks to @TalkMex. Pictures are very sharp.
Topic: Camera

Table 1: Sample Comment

Each dataset contains approximately 5000 tuples.

[1] https://github.com/dav009/LPForTopicIdentification
[2] http://merideangroup.com/

3 Pre-processing

The Datasets used in this experiment are actual data crawled from twitter and other sources, therefore there is a lot of noise in them.
Some of the problems are (but not limited) to:

- Miss spelled words, from typo errors to lack of accents.
- Internet Language, replacing some letters for others whose phonetics are similar i.e: "**qu**iero" (I want) for "**k**iero", abbreviations and expressions ("Jaja" "jiji" ...)
- Twitter Jargon such as: "RT:","@"..

The pre-processing done was the following:

1. Remove Strange characters, such as hearts, and other unicode characters
2. Remove some of the Twitter Jargon
3. Use TreeTagger [Sch94] for getting the part of speech and the stem of each word in each comment
4. discard those words that are not adjectives, nouns or verbs
5. replace each word by its stem in lowercase
6. remove accents from words

When it comes to pre-processing there could be a wide number of possible tasks that can be done, I try to keep it simple given the time constraints.

As a result of this pre-processing each comment is converted into a bag of keywords.
This will be later transformed into a vector space representation.

4 Label Propagation

This semisupervised learning graph method was proposed by Zhu [Zhu05].
The idea behind Label propagation is similar to K-Neighbours, nevertheless Label Propagation make use of the unlabelled data during the training process.

This approach maps the data to a graph representation. In this representation each arc of the graph will connect two nodes only if the two nodes are similar, the weights of the arcs are directly proportional to the similarity of the incident nodes.

With respect to our classification task,each node in the graph corresponds to the feature vector of a comment. The weights of the arcs are given by the cosine distance among the vectors.
The final representation is a complete graph.

In the LP algorithm, the label information of any node in a graph is propagated to nearby nodes through weighted arcs until a global stable stage is achieved [CJTN06]. If a weight arc is high, the label will travel easier through the graph.

The LP Algorithm iterates until convergence, at each step the algorithm will push the labels of the labelled data points through the arcs of high weights.The underlying assumption is that instances among a class will have high weighted arcs connecting them.
At each iteration the labels of the labelled data are clamped.

This propagation is done by measuring a transition matrix T. The transition matrix states the probability of propagating the label of an instance, and it is measured from the weights of the arcs of the graph.

Let T_{ij} be the probability of propagating the label of instance i to instance j, then it can be defined as:

$$T_{ij} = P(i \to j) = \frac{w_{ij}}{\sum_{k=1}^{n} w_{kj}} \tag{1}$$

4.1 Algorithm

Let Y_{ij} be the soft probability of labeling instance i with label j.

Given our semisupervised task, then Y can be divided into:

Y_U:the soft labels for the unlabelled data
Y_L:the soft labels for the labelled data

This means that Y_L is given at the beginning of the problem and the target is to find good values for Y_U.

Step 1 - Init: In this step, the labels for Y_L are clamped, and the labels for Y_U are randomly initialized.
The proof that the initialization values of Y_U are not transcendental can be found at [Zhu05].

Step 2 - Propagation: In this step, the labels of neighbouring nodes are pushed.
Let t be the current iteration, then Y^{t+1} is defined as:

$$Y^{t+1} = TY^t \tag{2}$$

Step 3 - Clamp Labelled Data: In this step, the Labels for Y_L are clamped.

Step 4 - Repeat: Repeat from step 2 until convergence.

Step 5 - Labeling Approach: Once there is convergence, a label has to be chosen for each node in the graph, there is more than one approach.
The simplest approach would be to pick the label with the highest probability.
In [Zhu05] there is an empirical analysis on more sophisticated approaches, there it is shown how this criteria can affect the performance, nevertheless for this experiment the simple approach was used.

More sophisticated versions of this approach have been presented and assessed, in [TP10] Talukdar and Pereira showed an empirical comparison among different variations of the LP algorithm, a modified absorption algorithm is presented where seed labels are not clamped,and it is shown to have better performance in the given tasks. Nevertheless within this paper we call LP to the algorithm proposed by Zhu [Zhu05].

To summarize in our experiment we convert our data into a complete graph, each node is the feature vector of a comment, and the weight of each arc corresponds to the similarity of the comments being incident to the arc.
This Graph is fed to the Label Propagation Algorithm proposed by Zhu.

5 Structure Learning

A semisupervised learning method proposed in [AZ05a].
This paper proposes a way to learn multiple classification tasks.In this approach the original Classification Task(Target Problem) is extended by creating a set of auxiliary classification tasks.
The auxiliary classification tasks are used to improve performance on the target task, the underlying assumption is that by solving the auxiliary problems one should be able to find good predictive structures.
The Auxiliary Tasks are trained on the unlabelled data,in consequence one constraint to the creation of auxiliary tasks is that they should be able to automatically generate training data from the target's problem unlabelled data.In [AZ05b] it is argued that having a high number of auxiliary tasks is beneficial for the performance.

In this paper's experiment, the target task corresponds to find the topic of each comment.
The auxiliary problems are defined as:
learning a linear predictor using the Unlabelled data for the K most common words in the unlabelled data and a predictor for the words with highest Pointwise mutual information(PMI) from the labelled Data.
By doing this, both the features of the labelled data and unlabelled data will be extended, the labelled data will be extended with features from the unlabelled data, and the unlabelled data will be extended with predictive features of the labelled data.

For learning this tasks, first the words with highest PMI are found using the labelled data, then the words with highest frequency in the unlabelled data are found.
Subsequently a linear classifier for predicting each of the most common words is trained using the unlabelled data, and a linear classifier for predicting each of the most predictive words is trained using the labelled data. It is clear that labels for this

tasks can be easily generated by masking the words for which the classifiers are being trained to predict.

By solving the auxiliary tasks, the feature vectors of the training data for the target problem can be extended with the predictions of the auxiliary tasks.Also the vectors of the unlabelled data are extended.
This might shown useful in the setup of the experiment, since we are extending the feature representations from knowledge of unlabelled data.

Finally a linear classifier is learnt using the training data for the target problem.
With respect to the ASO-SVD algorithm, this paper just explore a naive approach by using auxiliary problems, in this regard the auxiliary classifiers are used to extend the feature vectors.
For the rest of the paper we will refer to this approach as Naive SL.

6 Experiment

For the experiments, the three datasets mentioned above were used as training and test data.
The experiments corresponds to the following:

- Single task Supervised Learning using SVM (SVM)
- Single task semisupervised Learning using Label Propagation (LP)
- Single task semisupervised Learning using The naive Structural Learning(SL)

Each of the experiments were ran with different amounts of training data, specifically 10%, 30%, 60% and 90%

For the Label Propagation *Junto* [3] [TP10] was used. *Junto* is an open source implementation of the Label Propagation algorithm.

For SL and SVM the libsvm[4] library provided an implementation for training SVM.
For all the linear classifiers trained in the experiments a Polynomial Kernel was picked and the value of *gamma* was 1.2, the reason behind this choice is motivated by the empirical results obtained by [Joa98] in a similar task.

For making the tasks suitable for the given computational power, a subset of the size of $\frac{1}{4}$ of each of the given datasets was used, this has a major drawback, since the more the data the better the behaviour of the semisupervised algorithms. Furthermore an upper bound on the number of auxiliary classifiers created by the naive structure learning was imposed, the auxiliary problems would be 40% of the most common words in the unlabelled data and the words with highest PMI from the Labelled data.
A major drawback of Label Propagation from a running time perspective is having to calculate a complete graph containing the similarity among the nodes.Calculating a graph with 2 or more millions of arcs consume lots of resources.

7 Results

In overall the Naive SL could maintain the f-score given by the supervised approach, in some cases was able to get a much better f-score by improving recall, in some other cases a slightly better recall resulted in a decline of the precision thus leading to a worse f-score.

The LP approach on the other hand raised in labels where both of the previous approaches failed such as with the label *Resistance* in the sportswear dataset Figure 4 or *Brand Support* Figure 1, *Tenderness* in the Hygienic Product dataset Figure 6, and many others like *Photography*,*Screen* and *Social Networks* in the Mobilephone dataset Figure 12.

The Hygienic Product and Sportswear datasets shown some decent results, whereas the Mobile Phone dataset shown to be challenging.

One factor that might have affected the overall performance was using a subset of the data, this specially seems to affect LP.
LP profits from abundant unannotated data, my hypothesis is that using a very small set of seeds with a large number of unannotated data should

[3] https://github.com/parthatalukdar/junto
[4] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

improve the scores given below.
The reason for LP giving such low scores on some of the tasks might be caused by the limitation on the dataset sizes and the data sparsity in the feature vectors.

First the Limitation on the size of the datasets might have caused a lack of extra samples, thus generating some disconnected segments of the graph in case of LP, therefore a proper propagation could not be carried out successfully for prediction.

Second the feature vectors are very sparse, because of the bag of keywords approach and the short text nature of the task, thus leading also to disconnected segments of the graph.

A second test was carried out in an attempt to discover whether both more data and selecting more features would benefit the approach, this second test was done on the Mobile Phone dataset. In this trial more data were taken into account and a larger number of features were allowed to be part of the feature vectors,this would also benefit SL since the number of auxiliary problems would be incremented. The results are given in Figure 11 and Figure 12, in general it can be said that there is much a better f-score for a larger number of labels when compared to the previous tests, so It is possible to speculate that using the whole datasets would greatly increase the f-scores given in this report.

Since comments are reduced to a bag of keywords other factors which could have caused the given results are:

- **Noise**:the given data is actual user-generated content thus very noisy.
  Noisy data such as miss written words can lead to bad classifications or points isolated in the LP graph. Take as an example the label *Camera* for the Mobile phone dataset, if a good predictor for this label is the actual word "camera" and this word has been seen in the labelled data, seeing variations such as "camraa" might not result predictive, since they would be in a different dimension.
- **Sparsity and Short Text**: feature vectors are very sparse and text is short. This could affect in the following ways, lets think again on the label *Camera* for the Mobile phone dataset, if we assume we are classifying long texts, an annotated instance can reveal a lot of the topic vocabulary , in this case words such as "resolution", "megapixels" and so on. On the other hand in the short text classification, you got at most 144 characters per text, so you might end up only with a small subset of topic related words. For example lets assume you have a set of good predictors in your labelled data and those predictors are the words: "Camera" and "resolution".If there is a comment saying just: "it has a good amount of megapixels", it will be very unlikely to be classified correctly, unless there is another comment linking "megapixels" with one of the predicting words in the seed set.

  With respect to the topic vocabulary, I would speculate that this really affect the performance. In the Hygienic Product and Sportswear the term's range are more or less reduced, this is not the case of the mobile phone dataset, the range of topic vocabulary for each label is huge, just take into account the label *Application*, a comment can be given this label if it has a mention to a well known IOS or Android App, such as "Angry Birds", furthermore take into account the label *Social Network* where social networks' names are mentioned. On top of that combine all the possible technical words that might appear with the labels *Camera*, *OS*, *Photography*.

8 Conclusions and Future Work

An experiment on classifying short comments about products was described, two semisupervised approaches were presented and compared for solving this problem.

Data sparsity and the length of the texts shown to be an issue for Label Propagation. One idea for tackling this problem would be to try injecting some knowledge related to the labels, this could be useful in the difficult tasks such as the Mobile Phone dataset.An approach similar to the one given by [GM06] could be tried.Basically Wikipedia could be used to gather Topic vocabulary and use that vocabulary to extend each of the comments, this could solve disconnected segments in the graph.

Another idea worth trying is to use the SL for multitask classification instead of tackling each dataset by separate, it might be beneficial to join them and solve them as a single classification problem.However this requires more computational power but at the same time it should allow better results.

In spite of noise being a problem, I would speculate that given enough data the problem of noisy entries such as miss-spelled words should not be a big issue, this assuming there is enough noisy data in both the labelled and unlabelled entries.

It would be also interesting to try the other Label Propagation variations such as Modified Adsorption (MAD) discussed in [TP10], these implementations are available in Junto.
MAD should help when dealing with noisy input labels, which was one of the cases between *Photography* and *Camera* in the Mobile Phone dataset.

Finally considering joining the naive SL given here with LP should help the latter to get better scores, this by allowing to connect some of the isolated sub-graphs generated by the feature sparsity.

References

[AZ05a] Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.*, 6:1817–1853, December 2005.

[AZ05b] Rie Kubota Ando and Tong Zhang. A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 1–9, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[CJTN06] Jinxiu Chen, Donghong Ji, Chew Lim Tan, and Zhengyu Niu. Relation extraction using label propagation based semi-supervised learning. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 129–136, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[FH10] Xinghua Fan and Hongge Hu. A new model for chinese short-text classification considering feature extension. In *Proceedings of the 2010 International Conference on Artificial Intelligence and Computational Intelligence - Volume 02*, AICI '10, pages 7–11, Washington, DC, USA, 2010. IEEE Computer Society.

[GM06] Evgeniy Gabrilovich and Shaul Markovitch. Overcoming the brittleness bottleneck using wikipedia: enhancing text categorization with encyclopedic knowledge. In *proceedings of the 21st national conference on Artificial intelligence - Volume 2*, AAAI'06, pages 1301–1306. AAAI Press, 2006.

[Joa98] Thorsten Joachims. Text categorization with suport vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, ECML '98, pages 137–142, London, UK, UK, 1998. Springer-Verlag.

[Sch94] Helmut Schmid. Probabilistic part-of-speech tagging using decision trees, 1994.

[Sun12] Aixin Sun. Short text classification using very few words. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in* *information retrieval*, SIGIR '12, pages 1145–1146, New York, NY, USA, 2012. ACM.

[SUSB] Michael Speriosu, Sid Upadhyay, Nikita Sudan, and Jason Baldridge. Twitter polarity classification with label propagation over lexical links and the follower graph.

[TP10] Partha Pratim Talukdar and Fernando Pereira. Experiments in graph-based semi-supervised learning methods for class-instance acquisition. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1473–1481, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[Zhu05] Xiaojin Zhu. *Semi-supervised learning with graphs.* PhD thesis, Pittsburgh, PA, USA, 2005. AAI3179046.

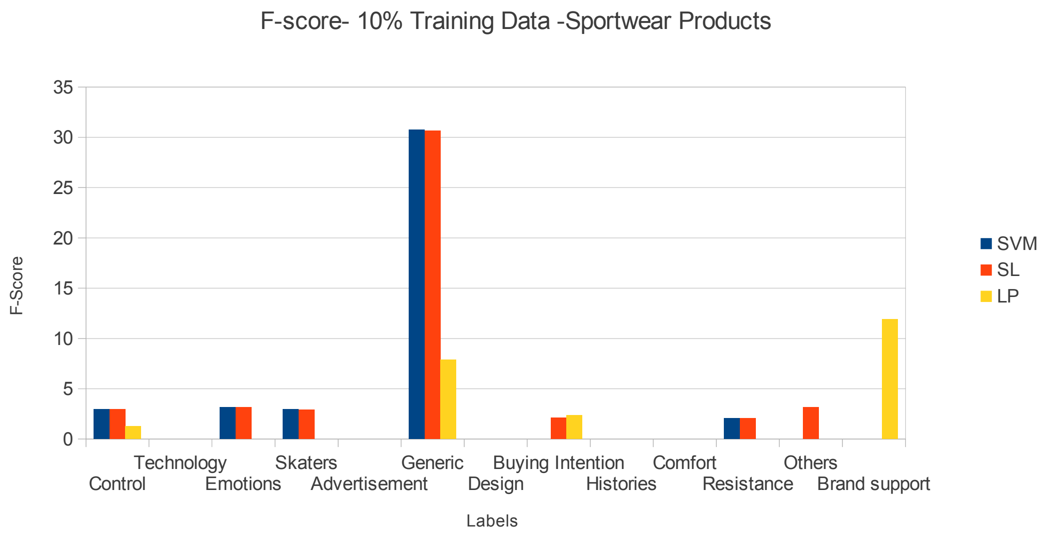

Figure 1: F-Score, 10% Training Data, Sportwear dataset

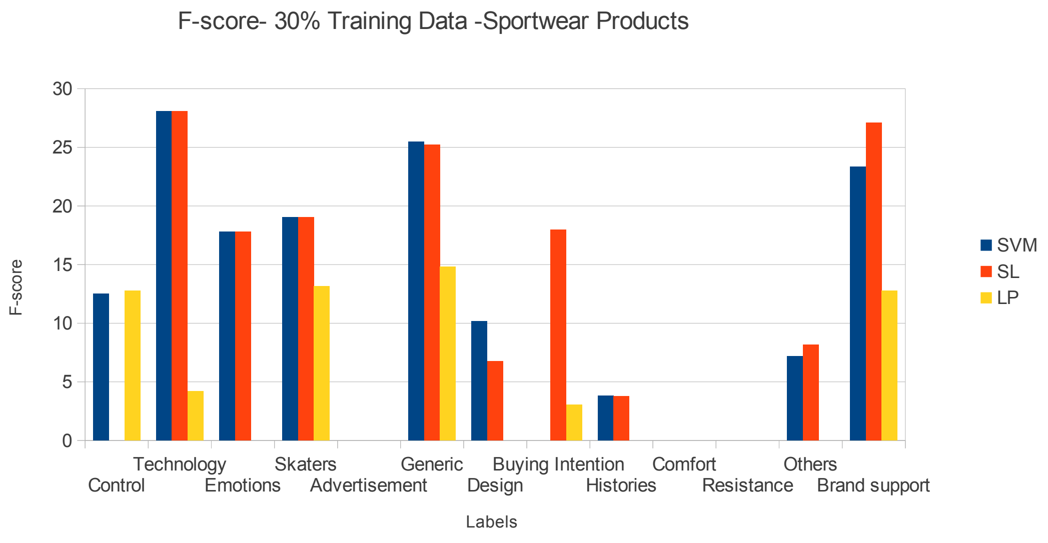

Figure 2: F-Score, 30% Training Data, Sportwear dataset

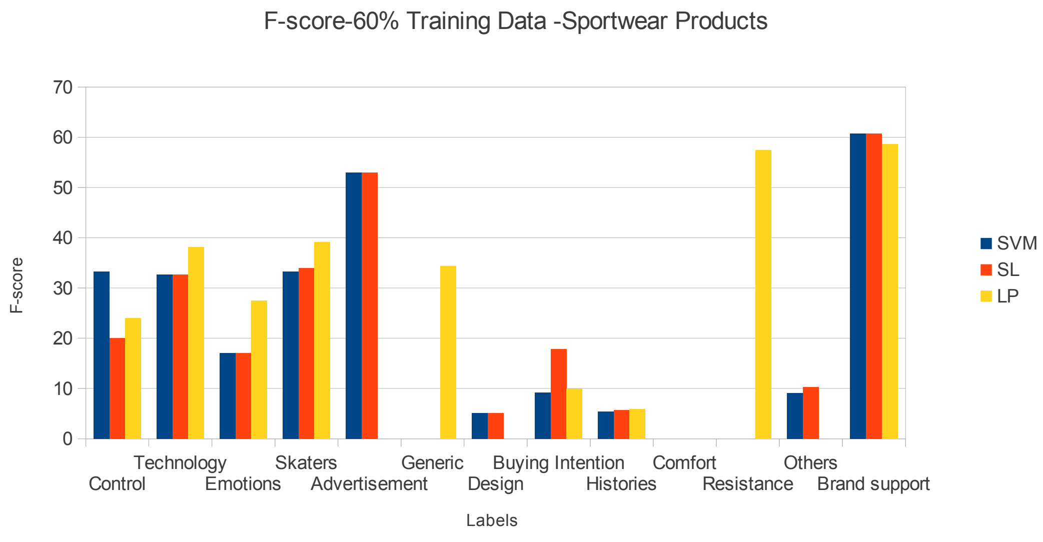

Figure 3: F-Score, 60% Training Data, Sportwear dataset

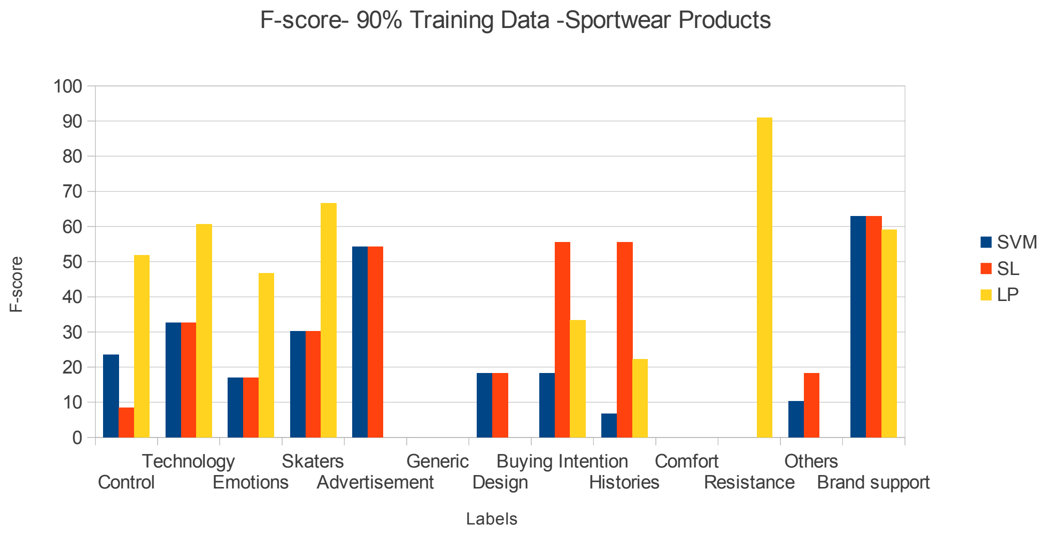

Figure 4: F-Score, 90% Training Data, Sportwear dataset

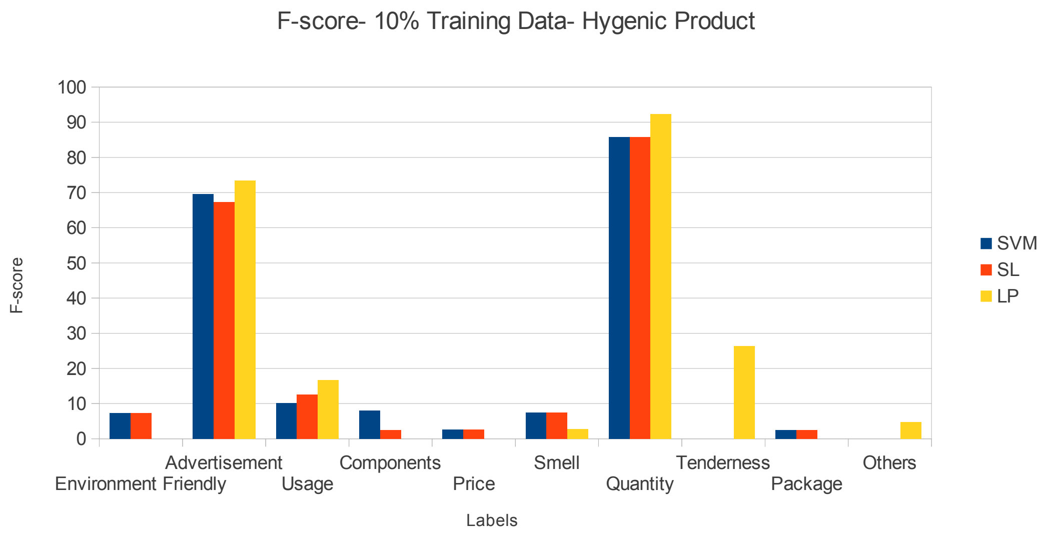

Figure 5: F-Score, 10% Training Data,Hygienic Product

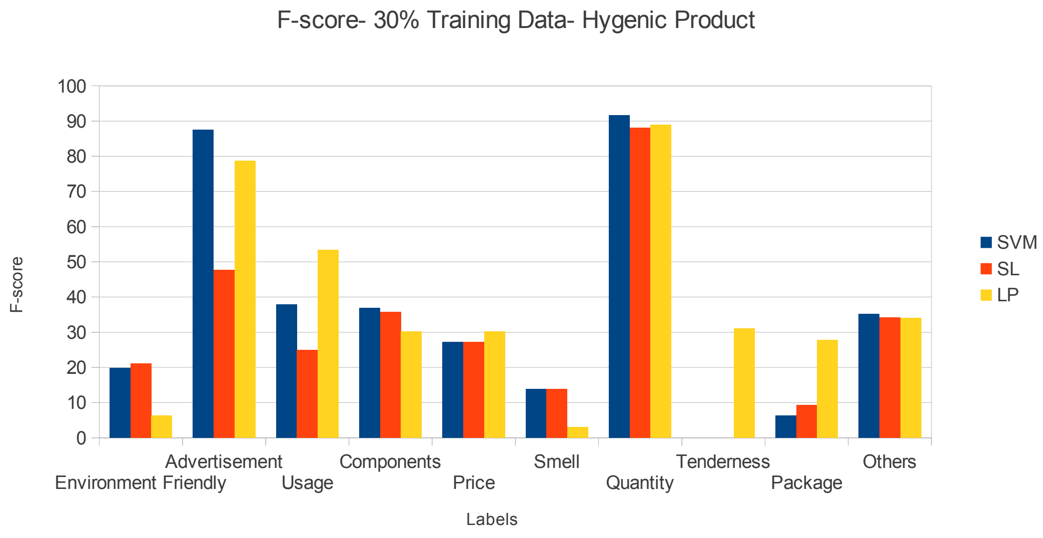

Figure 6: F-Score, 30% Training Data, Hygienic Product

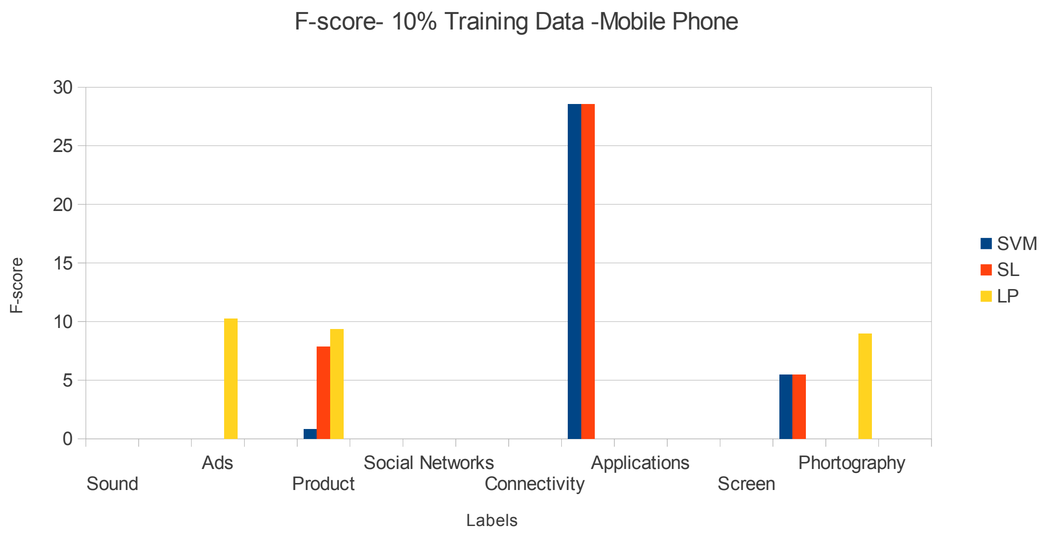

Figure 7: F-Score, 10% Training Data, Mobile Phone dataset

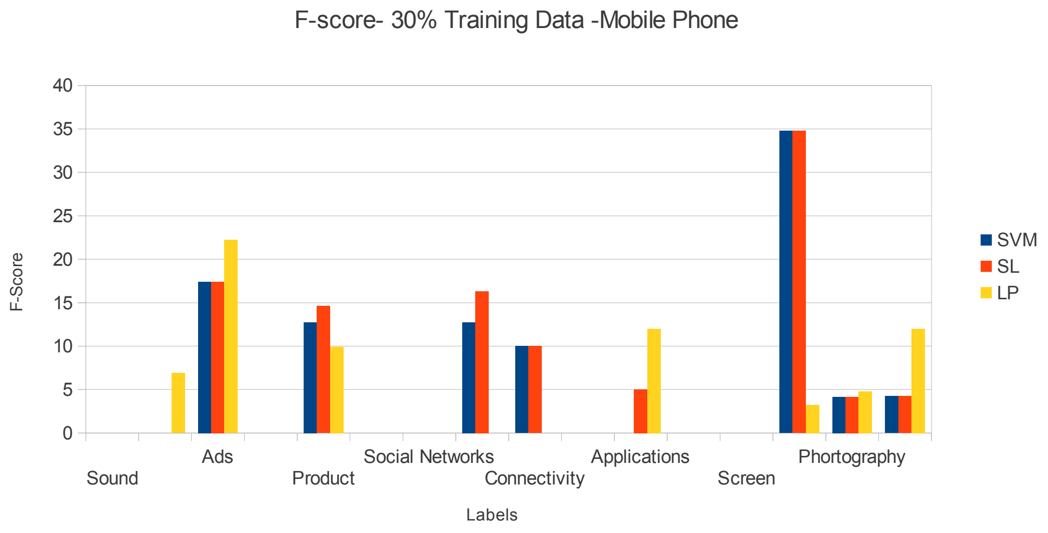

Figure 8: F-Score, 30% Training Data, Mobile Phone dataset

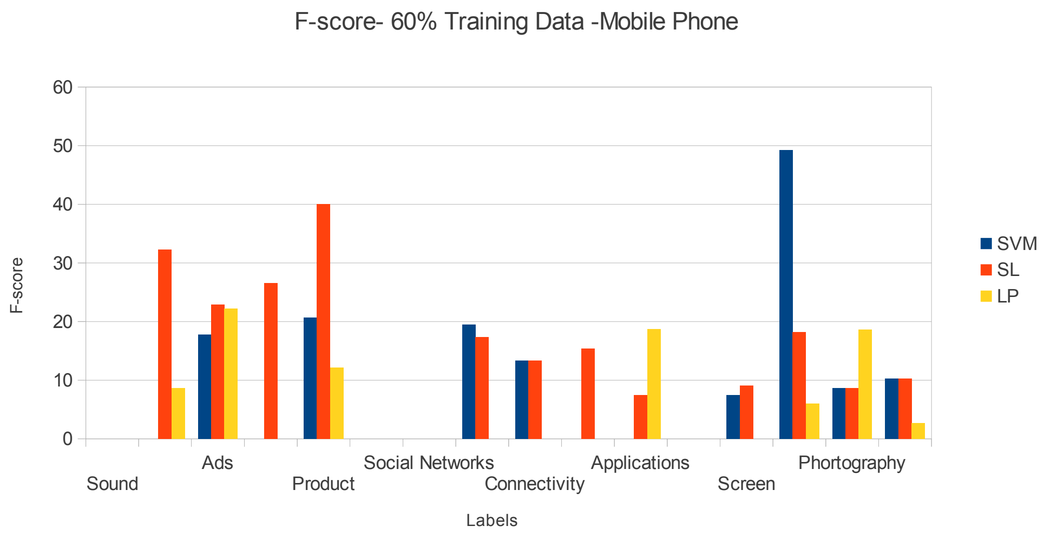

Figure 9: F-Score, 60% Training Data, Mobile Phone dataset

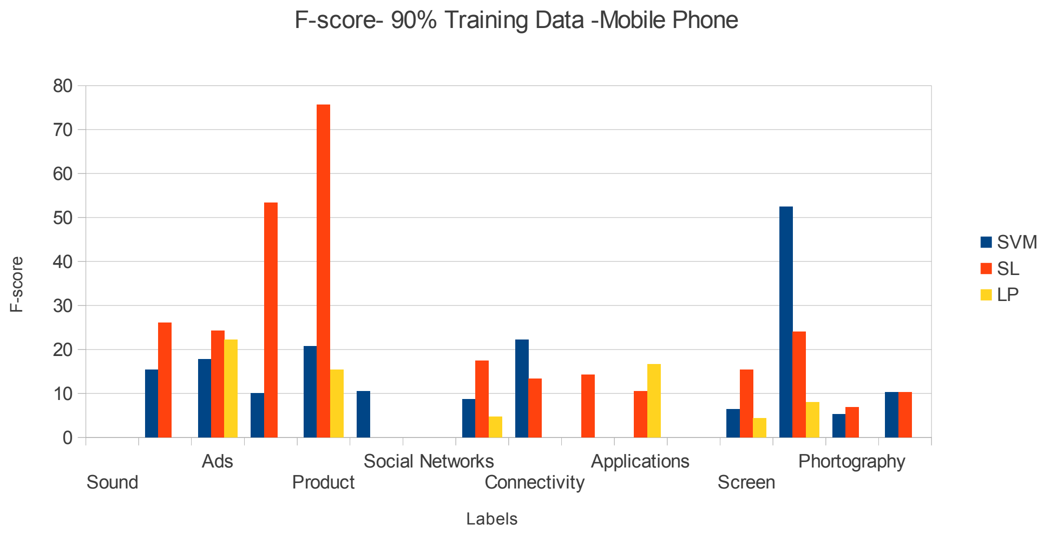

Figure 10: F-Score, 90% Training Data, Mobile Phone dataset

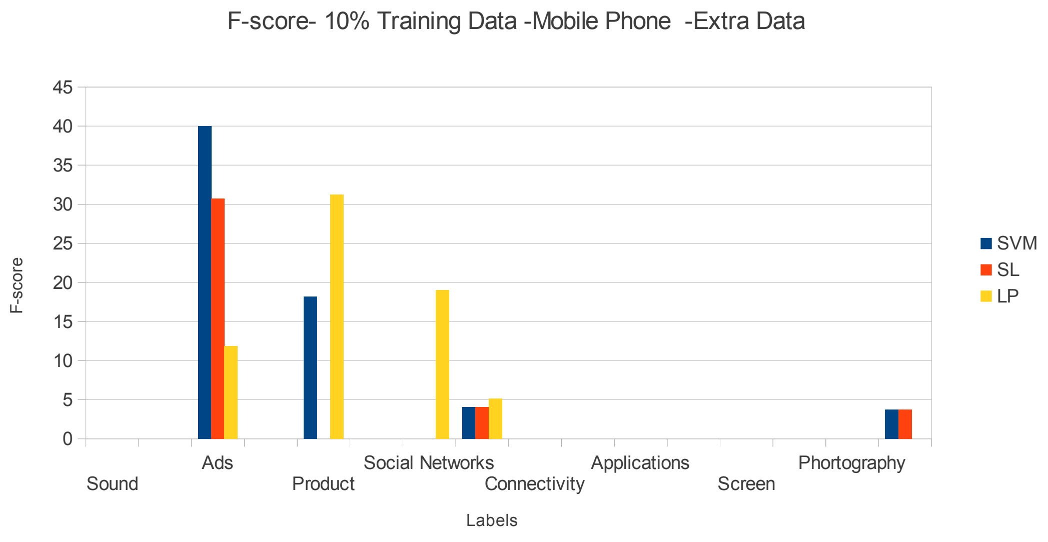

Figure 11: F-Score, 10% Training Data, Mobile Phone dataset, More Features and Data

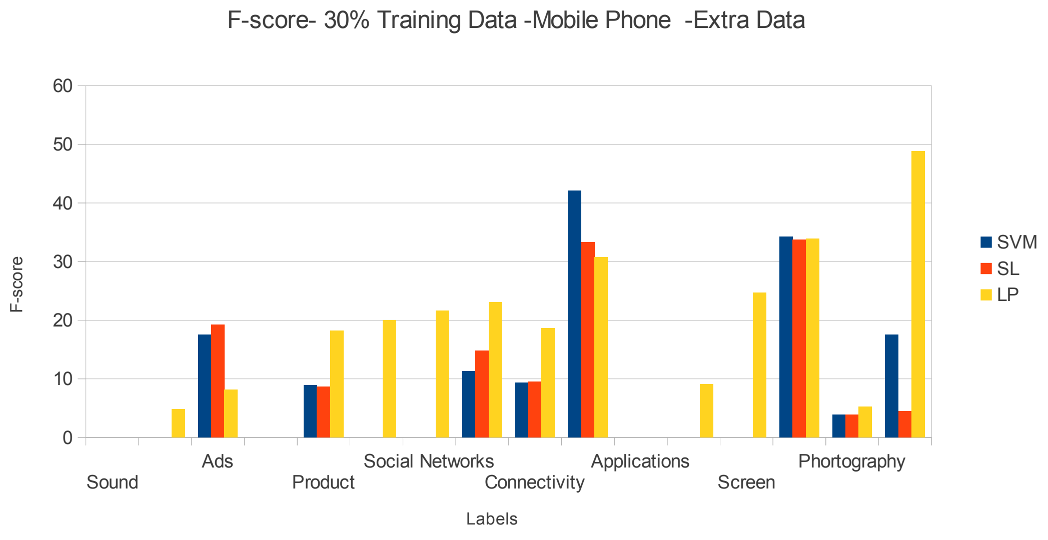

Figure 12: F-Score, 30% Training Data, Mobile Phone dataset,More Features and Data