Analysis of Comparative Study of TiMBL Algorithms over WSD

Vivek Patani (vpatani)
School of Informatics and Computing
Indiana University
Bloomington, Indiana 47404–0261
Email: vpatani@umail.iu.edu

Abstract—**This is an analytical study of various algorithms used on a neutral dataset to equally evaluate all the algorithms and study what suits what conditions better in order to thoroughly disambiguate the actual meaning of a word. The underlying software (framework) used to evaluate is Tilburg Memory-Based Learner. In this paper, we try to understand the algorithms, the evaluation table and what it infers.**

Keywords—TiMBL, comparative, contextual, word-sense, disambiguation.

I. Introduction

Lets̉ understand what does a Memory Based Learner mean? To simply put it in words, we can classify them as tools to find similarity between earlier and newer experiences. They can be divided into 2 functional components:

- **Learning Component** - This is the memory based part where in you *learn* the existing data and form a base case.
- **Performance Content** - This is the similarity based part where in you introduce a similarity measure and check how similar two data points are from the learning component.

II. Memory Based Learning

Before we delve into the anyltical comparison, we should just get a brief overview of the various similarity/dissimilarity measures along with the different algorithms provided TiMBL.

A. Distance Measures/Metrics

1) **Overlap Metric** – Simple sum of difference between two features. There are two variants in which it can be implemented, **Levenshtein** and **Dice** Coefficients. They have minor difference in implementation details.
2) **Information Gain & Gain Ratio Feature Weighing** – Infromation Gain reveals the isolated entropy of a feature between the two experiences. However, IG (Information Gain) fails to generalise and hence **Gain Ratio** is used, which normalise the IG by a factor of Split Info value.
3) **Chi Squared & Shared Variance Feature Weighing** – To overcome the drawbacks of Gain Ratio, where in feature selection is carried out based on statistical degrees of freedom. Shared Variance helps Chi Squared value to be more accurate.
4) **Modified Value Diff & Jeffrey Diversion Metrics** – Interestingly enough this is different from the all the methods mentioned above. MVDM (Modified Value Difference Metrics) is a method which calculates the co-occurence of features comparing it to the target class. Jeffrey Metrics is a measure of dissimilarity as compared to other metrics that we've seen.
5) **Dot - Products & Cosine Metrics** – When you have numeric or binary features, you can use a simple dot product. A Cosine Metric is once again a variant of the dot product.
6) **Distance Weighted Class Voting and Other Methods** – This is used as a base method for K Nearest Neighbours (Max Vote Method). Other methods consist of Tie - Breaking and Exemplar Weighting.

B. Algorithms/Indexing Strategies

1) **IB1** – As the size of the dataset increases, the access times can become more taxing and hence an Tree Based Indexing is used. IB1 is implemented using this idea.
2) **IGTree** – Based on the Information Gain, this is a different implementation where in the same tree from IB1 is in a compressed manner. IG is used as a parameter for the arc selection.
3) **TRIBL & TRIBL2** – To cover drawbacks from the IGTree and IB1 (or IB1-G), TRIBL & TRIBL2 came into existence. The simple difference between the two is that TRIBL fixes the point in feature ordering and TRIBL2 automatically selects that each classification.
4) **IB2 & Others** – This is a good strategy since each misclassification of kNN is recorded and kept in memory. The storage is much optimised, but fails at being disinterested while judging. It is fairly sensitive to noise.

III. Comparative Anaylsis

A. Statistical Insights

The above mentioned information will now guide us to analysing the data mentioned for Word Sense Disambiguation. Our first most or basic most observation is that while training, **Optimised training is always better** than the default set of training parameters. One more interesting pattern here is that while evaluating when you fine tune it, the accuracy almost always drops or remains equal but never better than as compared to the coarse tuning. Basic intuition would also indicate the same thing. If you observe carefully, you observe a large (or maximum pertaining to the table) difference between your training default setting and optimised setting for the words:

- **consume** – 25.9
- **bet as v** – 24.3
- **float as n** – 19.4

but while evaluation you see that there isn't as significant a difference between coarse and fine tuning compared to the training settings. However there is a significant peculiarity for the difference in accuracies of fine tuning and coarse tuning because all the words that appeared do not neccesarily repeat here (which means training data is not exactly telling you the accuracy measure):

- **giant as n** – 18.7
- **scrap as n** – 17.9
- **promise as n** – 14.2

B. Peculiarities

For any algorithm we know how important **robustness**[1] is. Robustness can be defined in *n* number of ways. In this case I would like to define it as the least amount of variance between the baseline and optimised parameters while training. The **k** that would be ideally used is 5 with **MVDM - IG**. The decision is based on the least amount of variance. Variance is simply calculated by subtracting average out of each of the differences (between baseline and optimal setting scores) and summing up all of the square of differences (or just add the differences). The results which turns out to be the lowest is the most robust since it will not be attenuated easily. The variance for MVDM IG is about 772 (0 if you simply sum). The next closest was MVDM IG with k as 1, with a variance of about 1000 (-0.2 if you simply sum).

If you observe, all the values of k are odd in nature (1,3,5,7,etc.). This is for a reason and not coincidence. It is recommended to keep k odd because each neighbour should have a majority and keeping it even may result in a tie and a majority would not come out. Hence, odd it is.

In general we know monosemous words are easy to disambiguate as compared to polysemous words. Words like **Giant** (as Adjective) have only one meaning (from its other form such as Noun), even the table depicts the same result. The word has a high baseline in itself and even the training & evaluation depict the very same. Now, a word like float (the Verb form) can be quite tricky to disambiguate. You can float on water and across air. Sentences in the dataset seem to be really close to each meaning and the table shows that the baseline is low and so is the overall accuracy. Just as a fact, words like *run* & *take* have over 100 meanings and can be utterly difficult to disambiguate since you need to test a lot of meanings before you drive yourself to a conclusion.

Looking at individual features maybe interesting but looking at the bigger picture gives you a few insights too. A 10 fold cross validation is carried out to avoid having a biased classifier and hence a biased output. I feel that Cross Validation neutralises the idea of difficulty that may occur during training or testing but nonetheless, we see as given in the table that training the dataset is difficult as compared to the easy evaluation set. The efforts required in improving baseline accuracy to training accuracy followed by optimisations is far more as compared to testing with the help of already given training set since tuning does not drastically improve the accuracy comparatively.

Acknowledgment

I would personally like to thank Prof. Cavar and Atreyee Mukherjee, they have guided me and provided me a lot of knowledge. I would also like to acknowledge the TiBML documentation, it has been a great source of information as well. Lastly, I would like to acknowledge my colleague, Raghu who also discussed with me and helped me understand certain concepts.

References

[1] Youmin Zhang, *A fast U-D factorization-based learning algorithm* , 1045-9227. Harlow, England: Canada, 2002.