SENSEVAL-3: Third International Workshop on the Evaluation of Systems
for the Semantic Analysis of Text, Barcelona, Spain, July 2004
Association for Computational Linguistics

Finding optimal parameter settings for high performance word sense disambiguation

Cristian Grozea
Department of Computer Science
University of Bucharest
Str. Academiei 14, 70109 Bucharest, Romania
chrisg@phobos.ro

Abstract

This article describes the four systems sent by the author to the SENSEVAL-3 contest, the English lexical sample task. The best recognition rate obtained by one of these systems was 72.9% (fine grain score) .

1 Introduction. RLSC algorithm, input and output.

This paper is not self-contained. The reader should read first the paper of Marius Popescu (Popescu, 2004), paper that contains the full description the base algorithm, Regularized Least Square Classification (RLSC) applied to WSD.

Our systems used the feature extraction described in (Popescu, 2004), with some differences.

Let us fix a word that is on the list of words we must be able to disambiguate. Let k be the number of possible senses of this word .

Each instance of the WSD problem for this fixed word is represented as an array of binary values (features), divided by its Euclidian norm. The number of input features is different from one word to another. The desired output for that array is another binary array, having the length k.

After the feature extraction, the WSD problem is regarded as a linear regression problem. The equation of the regression is $Aw = t$ where each of the lines of the matrix A is an example and each line of t is an array of length k containing $k - 1$ zeros and a single 1. The output xw of the trained model w on some particular input x is an array of values that ideally are just 0 or 1. Actually those values are never exactly 0 and 1, so we are prepared to consider them as an "activation" of the sense recognizers and consider that the most "activated" (the sense with highest value) wins and gives the sense we decide on. In other words, we consider the xw values an approximation of the true probabilities associated with each sense.

The RLSC solution to this linear regression problem is $w = A^t(AA^t + \lambda I_k)^{-1}t$;

The first difference between our system and Marius Popescu's RLSC-LIN is that two of the systems (HTSA3 and HTSA4) use supplementary features, obtained by multiplying up to three of the existing features, because they improved the accuracy on Senseval-2.

Another difference is that the targets t have values 0 and 1, while in the Marius Popescu's RLSC-LIN the targets have values -1 and 1. We see the output values of the trained model as approximations of the true probabilities of the senses.

The main difference is the postprocessing we apply after obtaining w. It is explained below.

2 Adding parameters

The obviously single parameter of the RLSC is λ. Some improvement can be obtained using larger λ values. After dropping the parser information from features (when it became clear that we won't have those for Senseval-3) the improvements proved to be too small. Therefore we fixed $\lambda = 10^{-8}$.

During the tests we performed it has been observed that normalizing the models for each sense (the columns of w) - that is dividing them by their Euclidian norm - gives better results, at least on Senseval-2 and don't give too bad results on Senseval-1 either. When you have a yes/no parameter like this one (that is normalizing or not the columns of w), you don't have too much room for fine tuning. After some experimentation we decided that the most promising way to convert this new discrete parameter to a continuous one was to consider that in both cases it was a division by $|w_i|^\alpha$, where $\alpha = 0$ when we leave the model unchanged and $\alpha = 1$ when we normalize the model columns.

3 Choosing the best value of the parameters

This is the procedure that has been employed to tune the parameter α until the recognition rate achieved the best levels on SENSEVAL-1 and 2 data.

1. preprocess the input data - obtain the features

2. compute $w = A^t(AA^t + \lambda I_k)^{-1}t$

3. for each α from 0 to 1 with step 0.1

4. test the model (using α in the post-processing phase and then the scoring python script)

At this point we were worried by the lack of any explanation (and therefore the lack of any guarantee about performance on SENSEVAL-3). After some thinking on the strengths and weaknesses of RLSC it became apparent that RLSC implicitly incorporates a Bayesian style reasoning. That is, the senses most frequent in the training data lead to higher norm models, having thus a higher aposteriori probability. Experimental evidence was obtained by plotting graphs with the sense frequencies near graphs with the norms of the model's columns.

If you consider this, then the correction done was more or less dividing implicitly by the empiric frequency of the senses in the training data. So, we switched to dividing the columns w_i by the observed frequency f_i of the i-th sense instead of the norm $|w_i|$. This lead to an improvement on SENSEVAL-2, so this is our base system HTSA1:

Test procedure for HTSA1:

1. Postprocessing: correct for i=1..k the model w_i by doing $w_i = w_i / f_i^\alpha$

For each test input x do 2,3

2. Compute the output $y = xw$ for the input x

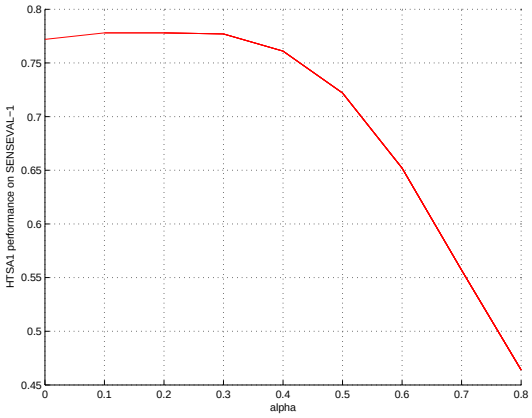
3. Find the maximum component of y. Its position is the label returned by the algorithm for the the input x.

Please observe that, because of the linearity, the correction can be performed on y instead of w, just after the step 2 : $y_i = y_i / f_i$. For this reason we call this correction "postprocessing".

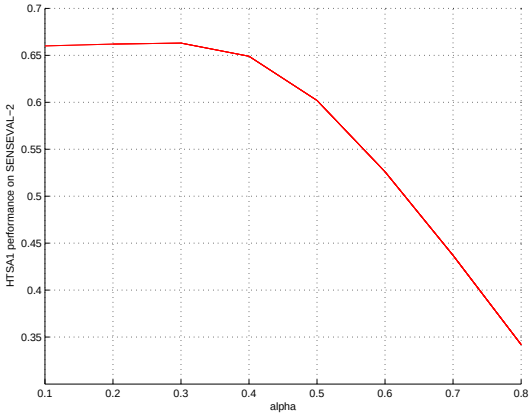
4 Description of the systems. Performance.

Here is a very short description of our systems. It describes what they have in common and what is different, as well which is their performance level (recognition rate).

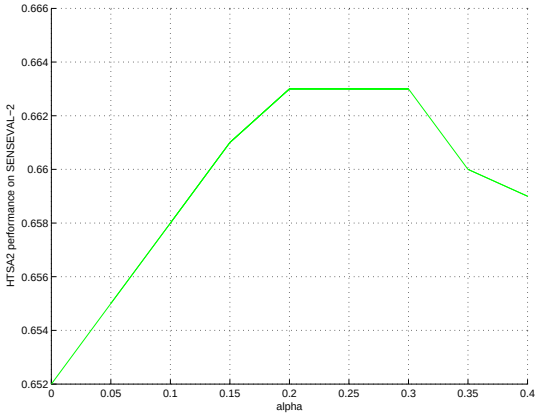
There are four flavors of the same algorithm, based on RLSC. They differ by the preprocessing and by the postprocessing done (name and explanation is under each graphic).



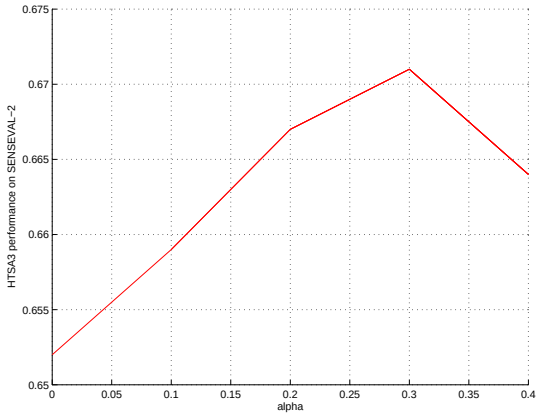
HTSA1: implicit correction of the frequencies, by dividing the output confidences of the senses by the $frequency^\alpha$; The graphic shows how the recognition rate depends on α on SENSEVAL-1.



HTSA1 on SENSEVAL-2 - the recognition rated depicted as a function of α

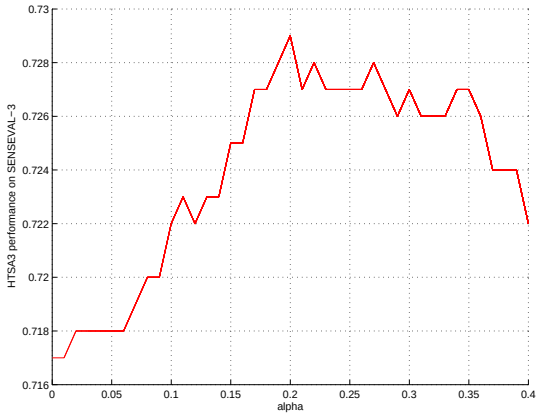


HTSA2: explicit correction of the frequencies, by multiplying the output confidences by a certain decreasing function of frequency, that tries to approximate the effect of the postprocessing done by HTSA1; here the performance on SENSEVAL-2 as a function of α.

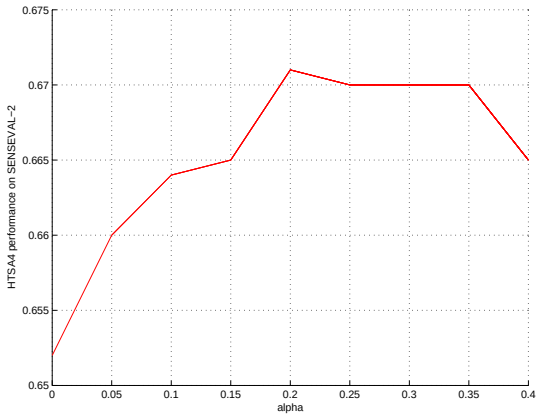


HTSA3: like HTSA1, with a preprocessing that adds supplementary features by multiplying some of the existing ones; here the performance on SENSEVAL-2 as a function of α.

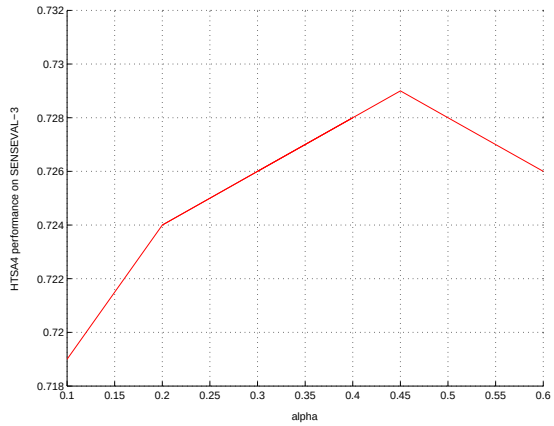
The supplementary features added to HTSA3 and HTSA4 are all products of two and three local context features. This was meant to supply the linear regression with some nonlinear terms, giving thus the algorithm the possibility to use conjunctions.



Was our best result lucky? Here is the performance graph of HTSA3 on SENSEVAL-3 as a function of α. As we can see, any α between 0.2 and 0.3 would have given accuracies between 72.6% and 72.9%.



HTSA4: like HTSA2, with the preprocessing described above. Here the performance on SENSEVAL-2 as a function of α.



The performance of HTSA4 on SENSEVAL-3 as a function of α.

What can be seen on this graphic is that $\alpha = 0.2$ was not such a good choice for SENSEVAL-3. Instead, $\alpha = 0.45$ would have achieved a recognition rate of 72.9%. In other words, the best value of α on SENSEVAL-2 is not necessary the best on SENSEVAL-3. The next section discusses alternative ways of "guessing" the best values of the parameters, as well as why they won't work in this case.

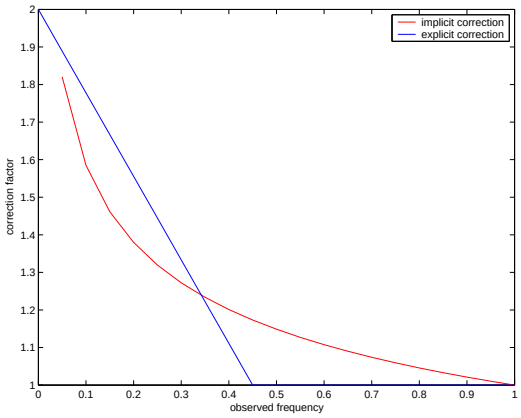
5 Cross-validation. Possible explanations of the results

The common idea of HTSA 1,2,3 and 4 is that a slight departure from the Bayes apriori frequencies improves the accuracy. This is done here by post-processing and works on any method that produces probabilities/credibilities for all word senses. The degree of departure from the Bayes apriori frequencies can be varied and has been tuned on Senseval-1 and Senseval-2 data until the optimum value $\alpha = 0.2$ has been determined.

Of course, there was still no guarantee on how good will be the performance on SENSEVAL-3. The natural idea is to apply cross-validation to determine the best α using the current training set. We tried that, but a very strange thing could be observed. On both SENSEVAL-1 and SENSEVAL-2 the cross-validation indicated that values of α around 0 should have been better than 0.2.

We see this as an indication that the distribution of frequencies on the test set does not fully match with the one of the train set. This could be an explanation about why it is better to depart from the Bayesian style and to go toward the maximum verosimility method. We think that this is exactly what we did.

Initially we only had HTSA1 and HTSA3. By looking at the graph of the correction done by dividing by $frequency^{0.2}$, reproduced below in red, we observed that it tends to give more chances to the weakly represented senses. To test this hypothesis we built an explicit correction, piecewise linear, also reproduced below on the same graphic. Thus we have obtained HTSA2 and HTSA4. In their case, α is the position of the joining point. Those performed close to HTSA1 and HTSA3, so we have experimental evidence that increasing the apriori probabilities of the lower frequency senses gives better recognition rates.



Red: Implicit correction (HTSA 1, 3); Blue: Explicit correction (HTSA 2, 4)

6 Conclusions. Further work.

RLSC proved to be a very powerful learning model. We also believe that tuning the parameters of a model is a must, even if you have to invent parameters first. We think that the way we have proceeded here with α can be applied to other models, as a simple and direct post processing. Of course the right value of α has to be found case by case. We would suggest everyone who participated with systems that produce Bayesian-like class probabilities to try to apply this postprocessing to their systems.

References

Marius Popescu. 2004. Regularized least-squares classification for word sense disambiguation. In *Proceedings of SENSEVAL-3*, page N/A.