Chapter 5

Concluding remarks

This chapter will start with a section that summarizes and compares the results of the experiments described in the earlier chapters of this thesis. After that we will describe studies performed by others that were inspired by our work. The chapter will be concluded with a section that presents the research tasks that we see as a possible follow-up on this thesis.

1 Experiment results

In this thesis we have described the application of machine learning techniques to the problem of discovering a phonotactic model for Dutch monosyllabic words. We have performed experiments with three learning algorithms, two data representation methods and two initialization schemes. The learning algorithms have only been provided with positive training data. Our goals were to find out which of the learning methods would perform best and to find out what data representation and what initialization scheme would enable the learning process to generate the most optimal model. The results of the experiments have been summarized in figure 5.1.

The first learning method we have examined was the statistical learning method Hidden Markov Model (HMM). We have used bigram HMMs which consider two characters at a time during the string evaluation process because regarding a context of one character is necessary for building a good phonotactic model. This learning method has produced good phonotactic models: after training the HMMs would accept around 99% of unseen positive test data and reject between 91 and 99% of the negative test data. There was only a small difference between training with random and initialized models but the models performed better with phonetic than with orthographic data. One observed difference was that the training process of linguistically

orthographic data	random initialization		linguistic initialization	
learning algorithm	% accepted positive data	% rejected negative data	% accepted positive data	% rejected negative data
HMM	98.9	91.0	98.9	94.5
SRN	100	8.3	100	4.8
ILP	99.2	60.0	97.8	97.7

phonetic data	random initialization		linguistic initialization	
learning algorithm	% accepted positive data	% rejected negative data	% accepted positive data	% rejected negative data
HMM	99.1	98.3	99.1	99.1
ILP	99.2	70.6	99.2	98.3

Figure 5.1: The results of our experiments with generating phonotactic models for Dutch monosyllabic words. The experiments included three learning algorithms, two initialization configurations and two data representations. No experiments have been performed with SRNs for the phonetic data representation because of the discouraging SRN results for the orthographic data representation. The HMM results for orthographic data with linguistic initialization come from the modified initialization (figure 2.19). The ILP results have been obtained with extended models (figure 4.8). The rejection scores for the negative phonetic data have been computed for the set of 574 incorrect phonetic strings.

initialized HMMs required significantly less time than that of the ones with a random initialization.

The connectionist method Simple Recurrent Network (SRN) was the second method that we have tested. This method performed surprisingly worse than the perfect results reported in (Cleeremans 1993). SRNs produced phonotactic models that accepted all unseen positive test data. However, none of the SRN models has been able to reject more than 8.3% of the negative test data. We have been able to show that the poor performance was caused by the large complexity of our training data. Characters in the data of Cleeremans et al. could be followed by at most two different characters while characters in our data can be followed by up to twenty characters. For the phonetic data this difference is even larger. Therefore we have refrained from performing SRN experiments with phonetic data.

The third learning method that we have looked at was the rule-based learning algorithm Inductive Logic Programming (ILP). This algorithm has generated good phonotactic models with linguistic initialization but the models generated with random initialization had problems with rejecting negative test strings. The large score difference for rejecting negative strings (on average 98% versus 65%) indicates that training with linguistic initialization enables this learning method to produce better models than training without this basic knowledge. We have performed two extra experiment groups with more elaborate rule formats and with rule set compression but these have not led to large performance differences.

It is easier to determine which of the learning methods has generated the worst phonotactic models than to point at a single method that has done best. The performance of the SRN models was much worse than the models generated by the other two methods because they failed to reject many negative test strings. HMMs generated better models than ILP in training processes without linguistic initialization. However when the algorithms were equipped with basic linguistic knowledge they would generate models which performed equally well. When it comes to choosing one of the learning methods for future studies, we would recommend using ILP for three reasons. First because ILP is capable of generating good phonotactic models when equipped with initial linguistic knowledge. Second because it, unlike HMMs, generates models that consist of rules which can be inspected and understood by humans. And third because the algorithm trained faster than its closest rival HMMs.

An inspection of figure 5.1 will reveal the answer to the question which data representation format, orthographic or phonetic, has suited the learning processes best. When we compare the results of the experiments with HMMs and ILP we see that in all cases the scores for the phonetic experiments are as least as good or better than the scores for the corresponding orthographic experiments. Although this is not a proof, it is an indication that it has been easier for the learning algorithms to discover regularities in the phonetic data than in the orthographic data. This result has surprised us. The larger number of different characters in the phonetic data and the larger entropy of this data had led us to the expectation that it would be more difficult to build a good model for the phonetic data than for the orthographic data.

The answer to the question whether starting the learning process from an initial linguistic model would enable the generation of better phonotactic models can also be found by inspecting figure 5.1. In the HMM and ILP results the scores of the initialized experiments are as least as good as the noninitialized experiments in all but one case (ILP: accepted positive orthographic test data).[1] The difference is largest for the ILP rejection rate of negative test data. This is an indication that initial basic linguistic knowledge will help learning algorithms to generate better phonotactic models. In the HMM experiments initial linguistic knowledge also sped up the training process. These results have been in accordance with what we had expected.

So HMMs and ILP have generated good phonotactic models but the SRN models performed poorly We favor ILP over HMMs because ILP trains faster and generates models which are understandable for humans. The results of our experiments indicate that representing the data in phonetic format and having access to basic linguistic information ables the learning processes to generate better phonotactic models.

[1] This exception may have been caused by a less suitable initial orthographic model.

2 Recent related work

The publication in (Tjong Kim Sang 1995) of the research results mentioned in chapter three of this thesis has led to follow-up research by others. In this section we will discuss work by Stoianov and Nerbonne with Simple Recurrent Networks (SRNs), work by Bouma with SRNs and Synchronous-Network Acceptors and work by Klungel with genetic algorithms.

(Stoianov et al. 1998) discusses a series of experiments in which SRNs were trained and tested on building phonotactic models for orthographically represented monosyllabic and multisyllabic Dutch words. With a different data set than ours, the authors have achieved an SRN performance that is better than any of the learning methods tested in this thesis: a total error of 1.1% on accepting positive monosyllabic test data and rejecting negative data. The results were obtained in a sequence of three experiment set-ups in which each set-up improved the performance of the previous one.

In the third experiment set-up the authors changed their word evaluation routine from a function similar to the Cleeremans measure (our measure 1 from section 3.1 of chapter 3) to an evaluation routine in which the word score was equal to the product of the character scores. This decreased the error rate of the SRN from approximately 3.5% to 1.1%. In our experiments the word evaluation measure used by (Cleeremans 1993) performed worst. We have suspected that the Cleeremans measure can be improved and this new result provides more empirical support for that suspicion.

In the second experiment group the authors switched from a stand-alone SRN training process to a parallel competitive training process. In this training process the networks are tested at different time points and networks that perform poorly are replaced. This technique is borrowed from the genetics algorithms field and was suggested by Marc Lankhorst (Lankhorst 1996). It helps the training process to get out of local minima and increases the possibility of finding an SRN that performs well. A disadvantage of this approach is that it requires supplying the network with negative information during the training process. This method was explicitly excluded in our learning experiments.

Using the competitive training process enabled the SRNs to go down from a total error rate of 7.5% to 3.5% on monosyllabic data. The set-up of experiments in the first group, which reached the 7.5% error rate, comes closest to our own experiment set-up. However there are three important differences between this first group and our own experiments. The first difference is that (Stoianov et al. 1998) have implicitly dropped our constraint that all training data has to be accepted. The error rate was obtained by choosing a word acceptance threshold which accepted as many positive data as possible while rejecting as many negative data as possible. We have experimented with this approach and reached an error rate of 16.3% at best (chapter 3, figure 3.16, SRNs, measure 3, 90%). Again the consequence of this approach is that to obtain the best threshold one has to make the network evaluate negative data.

The second difference between this group of experiments and our experiments was that the training data was weighted by frequency. Frequent words occurred more often in the training data. The number of times that a word appeared in the training data was equal to the logarithm of its frequency observed in a big text corpus. (Bouma 1997) has shown that incorporating frequency information in the training data will help SRNs to generate better phonotactic models.

The third difference was that negative data that was close to positive data was removed from the test data set. For the data evaluation the authors used the string distance function Levenshtein distance. The only strings that were allowed in the negative data set were strings that differed in two or more characters from any word in the positive data set. This restriction will simplify the task of the networks but we do not know how large the influence will be on the performance.

Stoianov and Nerbonne have provided empirical evidence for the fact that SRNs can be used as phonotactic models for Dutch. This is not something which we want to dispute. We have taken the same position as (Cleeremans 1993): we were interested in finding out whether SRNs can *learn* a good representation for the phonotactic data. The authors have shown that this question should be answered with yes. However, the question whether SRNs are able to build good phonotactic models from positive data only, remains unanswered.

(Bouma 1997) presents a study in which SRNs and Synchronous-Network Acceptors (Drossaers 1995) have been used for generating phonotactic models for Dutch monosyllabic words. In the SRN experiments he used similar techniques as Stoianov and Nerbonne in their initial group of experiments: including frequency information and dropping the constraint that all training data must be accepted. No limitations were put on the negative data but the positive data was restricted to the top 67% of a frequency ordered word list. With this approach Bouma's SRN obtained a combined error of 10.2% at best.[2] In his experiments SRNs that worked with data in which frequency information was incorporated performed better (at best an error rate of 10.2%) than SRNs that were trained with data without such information (at best 15.3%).

A Synchronous-Network Acceptor (SNA) is a biologically-plausible self-organizing neural network developed by Marc Drossaers (Drossaers 1995). It can be considered as a two-layer feed-forward network with links between the cells in the output layer. SNAs use two different variants of Hebbian learning during the training process. Bouma has performed three experiments with SNAs: one with orthographic data and two with phonetic data. This data did not contained frequency information. The experiment with orthographic data was a success: the SNA achieved a combined error rate of 2.1%. The experiments with phonetic data generated results which were worse: an error rate of 3.8% for locally encoded data and 28.1% for data encoded with phonetic features.

The results of the experiments of Bouma show that there are neural networks which can acquire good phonotactic models for Dutch monosyllabic words with positive training data only. It came as a surprise to us that in these experiments phonetic

[2] The best result was obtained with a threshold value which was determined by examining the performance of the SRNs on the test data. We feel that the test data should not be taken in consideration when determining the evaluation measure.

training data resulted in a worse performance than orthographic data. The 28.1% can probably be improved by using a different phonetic feature encoding. Bouma's results for his orthographic data set, which differs from ours, are better than the results obtained in any of our orthographic experiments.

(Klungel 1997) describes a series of experiments with genetic algorithms which generate models for the phonotactic structure of Dutch monosyllabic words. These experiments have been inspired by work on generating finite state automatons with genetic algorithms by Pierre Dupont (Dupont 1994). Klungel has used finite state models as phonotactic models which he has represented as so-called chromosomes in the genetic algorithms. In each experiment 30 models were submitted to a continuous modification process in an environment in which the best ones had the largest chance to survive. The genetic algorithm had access to positive and negative phonotactic data.

Klungel has performed experiments with two evolution methods and three fitness functions. The evolution method Individual Replacement performed best. Of the three fitness functions the lowest error rate was obtained with the one that used the average of the accepted positive data and rejected negative data as a model evaluation score (combined error rate of 8.5%). However this function did not generate models which performed consistently in the later phase of the training process.

The experiments performed by Klungel show that it possible to generate reasonable phonotactic models with genetic algorithms. We believe that even better results are possible with different genetic operators and a different initialization phase. However improving the operators and the initialization phase for this learning problem is a nontrivial task. One can imagine that they would benefit from having access to basic linguistic knowledge. A disadvantage of genetic algorithms is that it seems necessary to supply this learning method with negative data during training.

3 Future work

The work presented in this thesis and the studies discussed in the previous section have left some questions unanswered. These can be dealt with by performing follow-up research. This section will discuss possible directions for such research.

In section 2.3 of chapter 1 we have attempted to compute the complexity of our data set in order to predict the difficulty of our learning problems. We have taken a look at data entropy and the Chomsky grammar hierarchy in order to achieve that goal. Neither of the two methods was able to give an unambiguous answer to the question whether the orthographic data set or the phonetic data set was more complex. We would be interested in finding data complexity measures which are better suited for predicting the difficulty of learning problems. One of the measures that could be suitable is the Kolmogorov complexity used in (Adriaans 1992).

The work of Mark Ellison (Ellison 1992) has been discussed in section 3.1 of chapter 1. Ellison has put five constraints on his learning algorithms. Of our learning methods Inductive Logic Programming (ILP) comes closest to Ellison's goal: it satisfies four of the five constraints. The first constraint, learning algorithms should work in isolation, is not satisfied because we have trained ILP with monosyllabic data. This can be fixed by using multisyllable words as training and test data for ILP.

The experiments in this thesis have been performed with monosyllabic words rather than multisyllabic data in order to keep down the complexity of the learning problem. Now that we have shown that machine learning techniques can generate good phonotactic models for monosyllabic data, the next logical step is to test them on multisyllabic data. Others have already shown that some machine learning techniques can generate good phonotactic models for multisyllabic data (Stoianov et al. 1998). It would be interesting to apply other learning methods to this type of data as well.

Our work with Simple Recurrent Networks (SRNs) has generated a large response. The poor SRN performance reported by us has inspired many others to suggest and test modifications of the learning algorithm and the experiment set-up. One of the modifications that was suggested was incorporating information about the frequency of the words in the training data (Stoianov et al. 1998) (Bouma 1997). The experiments performed with frequency based training data have resulted in better phonotactic models than our experiments did. These results have been obtained with SRNs. We would be interested in finding out whether Hidden Markov Models could benefit from frequency information in the training data as well.

The SRN experiments by Stoianov, Nerbonne and Bouma have used negative data for determining optimal network acceptance threshold values. We suspect that these threshold values will also reject part of the training data (see figure 3.16 in chapter 3). This raises two interesting questions. First one could ask if every SRN experiment would benefit from disregarding part of the training data after the training phase. In other words: Can we improve the performance of SRNs on the test data by determining the acceptance threshold values from the best *x*% ($x < 100$) of the training data after training rather than from all training data? Our own experiments suggest that this *x* value would be around 90%. This leads us to the second question: Would such a cut-off value be the same for all SRN experiments? Finding a universal cut-off value would enable us to determine SRN acceptance threshold values by using training data only and relieve us from having to use negative data.

We would like to see other machine learning techniques applied to our data. One group of techniques which seems useful are the memory-based lazy learning algorithms used in the work of Walter Daelemans and Antal van den Bosch. The results reported for these learning techniques applied to phonological and morphological problems have been better than for decision trees and the connectionist method backpropagation (Van den Bosch et al. 1996). However, these learning methods require both positive and negative training examples. Perhaps it is possible to construct a clever problem representation for getting around this requirement.

In studies that follow-up our work there are two issues that should be taken care of. In the first section of this chapter we have compared the results of our learning experiments. We have been unable to give exact answers to our three research questions because the lack of statistical data. Most of our experiments have resulted in single test scores. It would have been better if they had generated average test scores with standard deviations. This would have made possible comparison statements which were supported with significance information.

Obtaining statistical data for experiment results requires repeating experiments several times. This might not always be useful. For example, performing our ILP experiments one more time with the same data would lead to the same results because our ILP training method is deterministic and uses a static initialization. Here we need an experiment set-up called 10-fold cross validation used in the work of Daelemans and others and originally suggested by (Weiss et al. 1991). In this experiment set-up the positive data is divided in ten parts and the training and test phase are performed ten times while each time a different data part is excluded from the training data and used as test data. We suggest that future experiments with our data be performed in this way to enable a statistically based comparison of the different experiments.

A second issue that should be taken care of in future experiments is consistent usage of the same data sets. In our work we have used the same training and test data for the experiments with the three learning methods. However (Stoianov et al. 1998) and (Bouma 1997) have used different data sets and this makes comparison of their results with ours difficult. In order to prevent this from happening in the future we will make our data sets universally accessible so that future experiments can be performed with the same data.[3]

Future work in applying machine learning techniques to natural language will not be restricted to generating phonotactic models for monosyllabic or multisyllabic words. One goal of our work has been to show that these techniques can be applied successfully to a small section of this domain. We hope that this thesis will provide inspiration for others to continue with new applications of machine learning techniques in larger parts of the natural language domain.

[3] Our data sets can be found on http://stp.ling.uu.se/˜erikt/mlp/