Estimating Markov Model Structures

Thorsten Brants

Universität des Saarlandes, Computational Linguistics
P.O.Box 151150, D-66123 Saarbrücken, Germany
thorsten@coli.uni-sb.de

In Proc. of ICSLP-96, Philadelphia, PA

ABSTRACT

We investigate the derivation of Markov model structures from text corpora. The structure of a Markov model is its number of states plus the set of outputs and transitions with non-zero probability. The domain of the investigated models is part-of-speech tagging.

Our investigations concern two methods to derive Markov models and their structures. Both are able to form categories and allow words to belong to more than one of them. The first method is *model merging*, which starts with a large and corpus-specific model and successively merges states to generate smaller and more general models. The second method is *model splitting*, which is the inverse procedure and starts with a small and general model. States are successively split to generate larger and more specific models.

In an experiment, we show that the combination of these techniques yields tagging accuracies that are at least equivalent to those of standard approaches.[1]

1. MOTIVATION

Generally, the structures of Markov models in the domains of speech recognition and part-of-speech tagging are fixed (manually pre-determined), and parameter estimation is restricted to probability estimation. An exception are models that use an automatic clustering algorithm to determine categories, which in turn determine the number of states and their outputs. But this still leaves out the transitional structure and, even more important, assigns each word to one and only one category. Very often is is useful to assign a word to more than one category, since the transitional probabilities for a word heavily depend on its category and can be very different for the different uses of a word.

On the one hand, n-gram-models as used in part-of-speech tagging make the implicit assumption that all words belonging to the same category have a similar distribution in a corpus. This is not true in most of the cases. On the other hand, several different categories will exhibit the same statistical distribution suggesting that they could be combined.

Section 2. describes model merging, a technique to derive more general models from specific ones as applied to part-of-speech tagging. Section 3. introduces model splitting, the opposite of model merging which derives more specific models from general models. Sections 4. and 5. demonstrate the benefits of combining these techniques.

2. MODEL MERGING

Model merging is a technique for inducing model parameters for Markov models from a text corpus. It was introduced in [Omohundro, 1992] and [Stolcke and Omohundro, 1994] to induce models for regular languages from a few samples, and adapted to natural language models in [Brants, 1995] and [Brants, 1996].

Model merging induces Markov models in the following way: merging starts with an initial, very specialized model. There is exactly one path in the model for each utterance in the corpus and each path is used by one utterance only. Each path is assigned the same probability $1/u$, where u is the number of utterances in the corpus. This model is named the *trivial model* M_{triv}. It exactly matches the corpus.

States of the trivial model are merged successively. Two states are selected and removed and a new merged state is added. The transitions from and to the old states are redirected to the new state, the transition probabilities are adjusted to maximize the likelihood of the corpus; the outputs are joined and their probabilities are also adjusted to maximize the likelihood.

The criterion for selecting states to merge is the probability of the Markov model generating the training corpus S. We want this probability to stay as high as possible. Of all possible merges (generally, there are $k(k-1)/2$ possible merges, with k the number of states) we take the merge that results in the minimal change of the probability. For the trivial model and u pairwise different utterances the probability is $p(S|M_{triv}) = 1/u^u$. The probability either stays constant or decreases. It never increases because the trivial model is the

[1] I am grateful to Brigitte Krenn and Christer Samuelsson for comments on an earlier version of the paper.

maximum likelihood model, i.e., it maximizes the probability of the corpus given the model.

Model merging stops when a predefined threshold for the corpus probability is reached. Some statistically motivated criteria for termination are discussed in [Stolcke and Omohundro, 1994].

For a part-of-speech tagging task, we need a modified merging procedure for the following reason. After merging states that emit the same word, but with different tags, the one with the smaller probability gets "overridden". Since we are looking for the tag sequence with maximal probability, the state will always choose the word/tag pair with higher probability and never the other one.

There are two solutions to this problem. (a) Forbid the merge of states that emit the same word with different tags. (b) Handle lexical and contextual probabilities separately, and apply merging to contextual probabilities only.

Alternative (a) was chosen in [Brants, 1995]. Approach (b) is used in this paper and described in the following section.

2.1. Context Merging

Merging states q_1 and q_2 in the original model merging procedure results in using the following probabilities for transitions originating in q_1 and q_2:

Contextual	Lexical
Before merging	
$P(q_i\|q_1), P(q_i\|q_2)$	$P(w_i\|q_1), P(w_i\|q_2)$
After merging:	
$P(q_i\|q_1$ or $q_2)$ for $i \neq 1, 2$ $P(q_1$ or $q_2\|q_1$ or $q_2)$ otherwise	$P(w_i\|q_1$ or $q_2)$

Instead of performing the full merge, we use the following restricted merge, that only combines the context:

Contextual	Lexical
Before merging	
$P(q_i\|q_1), P(q_i\|q_2)$	$P(w_i\|q_1), P(w_i\|q_2)$
After context merging	
$P(q_i\|q_1$ or $q_2)$	$P(w_i\|q_1), P(w_i\|q_2)$

After context merging, one can still determine the lexical probabilities for each of the states, and not the joint probability only for states q_1 and q_2. This better reflects the part-of-speech tagging task, where we have to assign a unique tag to each word. Context merging, as opposed to the original model merging, does not shadow tags for some words.

3. MODEL SPLITTING

Model splitting is the converse of model merging: a state is selected and divided into two new states. Transitions are redirected and the outputs are distributed among the new states.

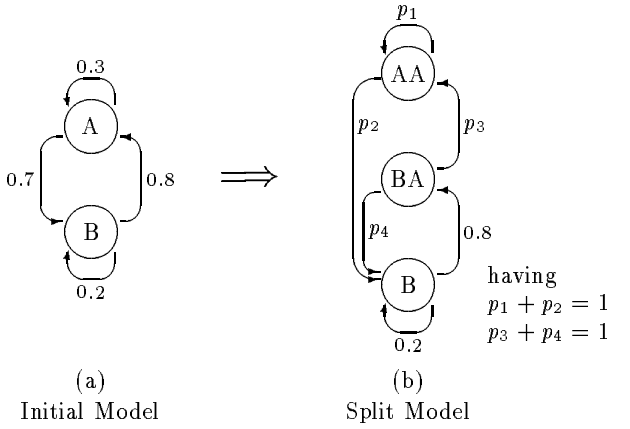

Figure 1: Context splitting for a first order Markov model. State A of the initial model is split into two states AA and BA (the first letter indicates the predecessor, the second letter the original state).

Two types of model splitting occur: context splitting and output splitting. In context splitting, the new state takes into account a longer part of the context than the original state, in output splitting the new states represent different categories of outputs. This investigation concentrates on the first type.

3.1. Context Splitting

Context splitting starts with some initial first order Markov model. A state of the model is selected that will be replaced by new states. The general case is to split a state into k new states. Since the same result can be achieved by $k-1$ splits into two states, the rest of this paper will deal with splits generating two new states.

The new states will take into account their predecessors, thus each of these new states represents a different extended context. Each transition into the original state is redirected to one of the new states, depending on the origin of the transition. All transitions from other states into the new states get the same probability as those originally ending at the replaced state. All new states inherit the set of transitions of the replaced state, but with different probabilities. These are estimated by maximum likelihood from the training corpus.

Figure 1 shows an example for context splitting. State A of the shown initial model is split into two states AA and BA. The first letters of the new states indicate their predecessors, the second letters indicate the proper states. BA means that the previous state was B and the actual state is A. This way the split model is again a first order Markov model. Each state of the initial model has transitions to each of the other states, including itself. This can no longer be true for the

split model. E.g., there cannot be a transition from state B to state AA, since AA represents state A with predecessor A. But when we transit from B to A, the predecessor is B. So there can only be a transition from B to BA, but not to AA.

Transitions originating in state B keep their probabilities, only the destinations have to be adjusted in those cases where the original transitions connected to the now split state.

The new states AA and BA inherit the complete set of transitions of the original state A. Transitions into the new states are adjusted according to the respective predecessor. The new probabilities $p_1 \dots p_4$ of these transitions are not completely determined by the initial model. By splitting a state we need more information for the model, and this additional information is determined by maximum likelihood from the training corpus.

The lexical probabilities are identical for all split states derived from the same parent.

In the case of splitting a state q_1, we use

Contextual	Lexical
Before splitting	
$P(q_i\|q_1)$	$P(w_i\|q_1)$
After context splitting	
$P(q_i\|s,q_1),\ P(q_i\|\bar{s},q_1)$	$P(w_i\|q_1)$

where s is some subset of states, and $\bar{s}$ its complement.

3.2. Finding Good Splits

Model splitting tries to find "interesting" information sources, i.e., information sources for which the conditional probability considerably vary from those which use the smaller context only, and for which enough data exists to make reliable estimates.

The quality of splits is quantified by the divergence Div of the resulting probability distributions, which is based on the relative entropy (or Kullback-Leibler distance) D:

$$D(P||Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

$$Div(P,Q) = D(P||Q) + D(Q||P) \quad (1)$$

for two probability distributions P and Q. The divergence is chosen instead of the relative entropy for its symmetry. The best split maximizes the divergence, thus we are looking for

$$\operatorname*{argmax}_s Div(P(\cdot|s,q_1), P(\cdot|\bar{s},q_1)). \quad (2)$$

Unfortunately, there are 2^k subsets s of a set with k states $\mathcal{Q} = \{q_1 \dots q_k\}$, thus there are 2^{k-1} hypothetical splits of which we want to find the best one.

One solution to this problem is proposed by [Schütze and Singer, 1994], who take a similar approach. They start with a model containing only one state. Each split results in two states, one of them represents one additional category in the context, the other one represents the rest of the categories. Expressed with equation (2), this means that the subset s is restricted to contain exactly one state. This misses a lot of splits that would improve the model, and the accuracy of their resulting model is lower than for standard approaches.

Therefore, we do not restrict s to contain only one state. Since the exponential growth of hypothetical splits makes the computation of the global maximum impossible, we use the following heuristic to derive a local maximum.

Let q_a and q_b be states that maximize

$$\max_{q_i,q_j} Div(P(\cdot|q_i,q_1), P(\cdot|q_j,q_1)).$$

These two states are assumed not to both be in s because of the high divergence of their respective probability distributions. Starting with these two states, we build two sets, one of which contains states yielding probability distributions that are closer to q_a, the other one containing states yielding distributions closer to q_b.

Without loss of generality, q_a is put into s (and q_b into $\bar{s}$). For all other states $q_k \in \mathcal{Q}$, $(q_k \neq q_a, q_b)$, we determine

$$\min_{q=q_a,q_b} Div(P(\cdot|q_k,q_1), P(\cdot|q,q_1)).$$

If the minimum is reached by q_a, then q_k is added to s, otherwise to $\bar{s}$. At the end, the split is performed on the basis of s and its complement.

4. COMBINING MERGING AND SPLITTING

When creating a language model of fixed order from a corpus (e.g., a bigram model), one can observe that some of the states have much more occurrences in the corpus than are needed for reliable probability estimates of their transitions, while others occur so seldom that their probability estimates are very unreliable.

The model is improved by specializing those parts for which there is sufficient data for splitting, by generalizing those parts for which there is insufficient data, and by leaving the rest untouched. Generalizing is done by model merging, specializing by model splitting. This results in the algorithm shown in figure 2.

We use a data driven approach to determine which states to merge and which ones to split. The states are sorted by their frequency. Beginning with the lowest frequency, states are merged, and beginning with the highest frequency, states are split. An extra held out part of the training data is used to determine the accuracy of the resulting model. The procedure stops, if the accuracy decreases for all possible merges (splits, resp.) for a state.

```
procedure EstimateStructure
begin
    M_0 := initial model of training corpus
    i := 0
    Q_low := Set of states with insufficient data
    while Q_low not empty do
        L_i := Set of merges with at least
            one element of Q_low
        Calculate corpus probs for merges in L_i
        M_{i+1} := Best merge in L_i performed on M_i
        i := i + 1
        Adjust Q_low
    end
    Q_high := States with more than enough data
    while Q_high not empty do
        q := a state from Q_high
        s := Subset of states for the split of q
        M_{i+1} := M_i, q split w.r.t. s
        i := i + 1
        Adjust Q_high
    end
    output M_i
end
```

Figure 2: Algorithm for estimating the transitional structure of a Markov model.

5. AN EXPERIMENT

This section reports on the application of merging and splitting for a part-of-speech tagging task. We used the Susanne corpus [Sampson, 1995] for the experiment. The tagset consists of 424 tags. The corpus was divided into four disjoint parts, one large training part (part A, approx. 130,000 words), and three smaller test parts (parts B – D, approx. 10,000 words each).

Tagging results of standard approaches are shown in table 1 (a)–(d). In all cases, unknown words are handled by analyzing their suffixes [Samuelsson, 1996]. The pure uni-, bi-, and trigram model (columns (a)–(c) of the table) use expected likelihood estimation for smoothing (i.e., constant addition of 0.5 to each frequency). Column (d) uses a linear interpolation of uni-, bi-, and trigrams, the weights are derived by deleted interpolation [Brown *et al.*, 1992].

As one can see in table 1, the results of the bigram model (b) are already very close to the linear interpolation (d), yielding the best accuracy of these models. Therefore, we start with a bigram model and apply merging and splitting.

The bigram model consists of 424 states (one state for each category). Merging was performed for the low frequency states up to frequency 87 (264 states), splitting was performed for the high frequency states down to frequency 1368 (24 states). The resulting model has 184 states for its transitional structure.

Table 1: Tagging results with standard models. Smoothing for (a), (b), and (c) is done by expected likelihood estimation (ELE), (d) uses a linear interpolation of (a), (b), (c) which is derived by deleted interpolation. (e) is a bigram model with estimated transitional structure.

	(a)	(b)	(c)	(d)	(e)
part	unigrams	bigrams	trigrams	linear int.	est.struct.
B	89.83%	94.49%	92.48%	94.90%	95.01%
C	89.80%	94.08%	92.66%	94.20%	94.28%
D	88.62%	93.17%	91.61%	93.48%	93.47%
Avg	89.41%	93.91%	92.25%	94.19%	94.25%

The tagging results using this model are shown in table 1(e). The are slightly, but not significantly better than those for linear interpolation.

6. CONCLUSION

We introduced model splitting, and a modification of model merging. Together, these two methods provide a powerful way to estimate the structure of a Markov model. We concentrated on the transitional structure of the model.

An experiment with a part-of-speech tagging task showed that the combination of model merging and model splitting is at least as powerful as linear interpolation of n-grams, which is a state-of-the-art technique.

7. REFERENCES

[Brants, 1995] Thorsten Brants. Estimating HMM Topologies. In *Tbilisi Symposium on Language, Logic, and Computation*, Human Communication Research Centre, Edinburgh, HCRC/RP-72, 1995.

[Brants, 1996] Thorsten Brants. Better Language Models with Model Merging. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, Philadelphia, PA, 1996.

[Brown *et al.*, 1992] P. F. Brown, V. J. Della Pietra, P. V. deSouza, J. C. Lai and R. L. Mercer. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, 1992.

[Omohundro, 1992] Stephen M. Omohundro. Best-First Model Merging for Dynamic Learning and Recognition. In J. E. Moody, S. J. Hanson and R. P. Lippmann (eds.), *Advances in Neural Information Processing Systems 4*, pages 958–965. Kaufmann, San Mateo, CA, 1992.

[Sampson, 1995] Geoffrey Sampson. *English for the Computer.* Oxford University Press, Oxford, 1995.

[Samuelsson, 1996] Christer Samuelsson. Handling Sparse Data by Successive Abstraction. In *Proceedings of COLING-96*, Kopenhagen, Denmark, 1996.

[Schütze and Singer, 1994] Hinrich Schütze and Yoram Singer. Part-of-Speech Tagging Using a Variable Memory Markov Model. In *Proceedings of ACL-94*, Las Cruces, NM, 1994.

[Stolcke and Omohundro, 1994] Andreas Stolcke and Stephen M. Omohundro. *Best-first Model Merging for Hidden Markov Model Induction.* Technical Report TR-94-003, International Computer Science Institute, Berkeley, California, USA, 1994.