Data Pruning for Template-based Automatic Speech Recognition

Dino Seppi & Dirk Van Compernolle

ESAT, Katholieke Universiteit Leuven, Leuven, Belgium

dino.seppi@esat.kuleuven.be, dirk.vancompernolle@esat.kuleuven.be

Abstract

In this paper we describe and analyze a data pruning method in combination with template-based automatic speech recognition. We demonstrate the positive effects of polishing the template database by minimizing the word error rate scores. Data pruning allowed to effectively reduce the database size, and therefore the model size, by an impressive 30%, with consequent benefits on the computation time and memory usage.

Index Terms: pruning, template, automatic speech recognition

1. Introduction

Statistical methods for pattern recognition rely on large amounts of data. In the field of Automatic Speech Recognition (ASR), experience has so far suggested that the more data are available, the better the models and the performance of the system: "there is no data like more data" cf. [1]. However, huge amounts of data are difficult to handle as they require plenty of memory for storing and computational power for training. Moreover, very large datasets are often automatically annotated so that the quality of transcriptions and segmentations could be severely affected. Therefore, from one side, corrupted data can be compensated by collecting more examples; on the other side, these new examples can introduce new (possibly fewer, in percentage) errors. This is perfectly manageable unless the model grows with the data—even linearly only: this is the case of Template-based Automatic Speech Recognizers (T-ASRs). Differently from architectures based on Hidden Markov Models (HMM), a T-ASR consists of a huge collection of acoustic realizations (referred to as episodes, examples, or templates) which are labelled, indexed, and stored in memory. For the recognition, the input speech is compared, using distance measures, with the templates. The best template sequence is the one that minimizes the global distance and possibly other additional costs. It is therefore evident how daunting a huge increment of the database could be, both w.r.t. the model size (i.e. the template database) and the decoding time and complexity.

In the past, so called data pruning (or cleaning, or selection) techniques were applied to cope with redundant, inaccurate, or even completely noisy data. These techniques have been investigated to improve the training phase of HMM-based speech recognizers. Successful results were obtained for cleaning the unsupervised part of speech databases [2]. More recently, it has been proven that a substantial reduction of the whole training material can be achieved [3]. Nevertheless, data cleaning has not been very popular for HMM, for a number of reasons. First, data cleaning can be evaluated only by retraining the HMM: this is computationally prohibitive. Second, HMMs rely on a compact representation of the training data; on the one hand, most of the phonetic details are lost, but on the other hand the parametric data representation makes HMMs more robust to noise and outliers. For instance, unreliable transcriptions or segmentation errors are averaged by the Gaussain models so that their influence is practically nullified by the overwhelming amount of correct data. At the same time, techniques such as state tying cope with data skewness: in less populated classes outliers could play a more important role. Finally, a more general reason for the limited popularity of data pruning could be that suspicious patterns may not always be garbage patterns as noisy data too might be needed to make the classifier learn difficult examples. Although ambiguous, many patterns often prove to be important and do characterize class distributions [4, 5].

As anticipated above, data pruning might be more useful if not absolutely necessary for a T-ASR. As all single examples have to be stored and checked against the input patterns to be recognized, we face problems of memory allocation and waiting time during recognition. A smaller, reliable subset of data has the primary advantage of easing and speeding computation. Eventually, spared resources could be dedicated to important and time demanding tasks such as the fine tuning of the model's parameters, which can eventually lead to a performance improvement too. Redundant patterns are good candidates to be pruned; they consist of groups of examples that can be highly inter-correlated: any additional, redundant example does not add, in general, useful information that can be successfully exploited by the recognizer. Also discarding templates responsible of errors might be beneficial. In our framework, the two more common sources of errors are the presence of outliers and of noisy segmentations; either source is probably more challenging to cope with than for HMMs, above all in a high dimensional feature space as the one we are working on [6]. Outliers are macroscopic errors mainly introduced during the annotation phase: the collection of large amounts of data to be manually labelled inevitably implies a certain number of wrong transcriptions. These examples, lying at the edges of class distributions, might be particularly detrimental as they can compromise the matching heavily. Moreover, the forced alignment needed to construct the template database is responsible in large part for fine-grained segmentation errors. Both these types of error sources can be tackled by data pruning methods.

In this paper we present an iterative pruning strategy that eliminates more and more patterns from an initial database; at each iteration feedback is provided by the WER of the current model. The aim is to discard both noisy and redundant examples. Noisy patterns can be spotted by minimizing the WER on left-out data; as immediate side effect, redundant patterns too can be isolated by the down-sampling as long as the WER is non-increasing. Such an approach could be possible only thanks to the nature of the T-ASR: no re-training of the models is needed after each iteration, because the model coincides with the (pruned) template database.

This work has been supported by the European Community under grant RTN-CT-2006-035561 (Sound-to-Sense) and by the Fund for Scientific Research Flanders, FWO, under grant G.0260.07 (TELEX).

Table 1: *The Wall Street Journal database: training set (*WSJ*) and evaluation sets (*dev.92 *&* eva.92*).*

data-set	spk.	hours	words	sent.	phon.
WSJ	284	81	644k	38k	2.8M
dev.92	10	0:34	6724	403	21.1k
eva.92	8	0:30	5643	333	16.7k

In Sec. 2 we describe the data that were used to train the HMM models and the template database. The system structure is drawn in Sec. 3 and the data pruning algorithm in Sec. 4. Finally, in Sec. 5 we present the results obtained so far and in Sec. 6 we conclude with a discussion on this and future work.

2. Data

The Wall Street Journal (WSJ) continuous speech recognition training corpus, consisting of almost 80 hours of speech, was chosen as training material for the HMM models. For testing, we opted for the $20k$ open vocabulary test set (eva.92) which is evaluated in combination with bi- and tri-grams language models. The dev.92 dataset was used to optimize the LM weights and word insertion penalty parameters used for the decoding. More details on the database are sketched in Tab. 1.

The template database was obtained by forced aligning the WSJ training data using triphone units. This yielded 2.8 millions of templates. In total we count 4264 template (or triphone) classes populated by (approximately) 150 up to 12000 (for a schwa allophone) different speech realizations.

3. System

The template-based ASR used for this work is a 2-layer recognizer. In the first layer, Word Graphs (WGs) are generated using classic acoustic and language models. In the second layer, the WG arcs are expanded into triphones, then the WGs are optimized, and finally triphones are mapped onto triphone templates. Detailed information on the system can be found in [7], however a few differences have been introduced for the purpose of this work. The main difference is that the second layer is characterized by a pure template matching. Moreover, to ease the computational burden, we skipped the application of additional costs to the graph, such as the *natural successor cost*, cf. [8]. To illustrate the quality of the baseline system, in Tab. 2 we present recognition results on the dev.92 and eva.92 datasets and compare the T-ASR with SPRAAK, an HMM-based state-of-the-art ASR.

In the **first layer**, WGs are generated using our in-house ASR engine, SPRAAK [9]. SPRAAK front-end extracts 72-dimensional feature vectors: 24 Mel-based coefficients with first and second order derivatives. VTLN and a decorrelation algorithm (MIDA, a generalized LDA) are later applied so that the final feature vector can be reduced to 36 features. The acoustic models are triphone based: $15.8k$ cross-word triphones models trained on the entire WSJ database share $3.4k$ states and each state distribution choses from a pool of $32.7k$ diagonal covariance Gaussians. Using these models we were able to build WGs from the training material, and from the dev.92 and eva.92 evaluation sets. In Tab. 2, we present the most salient characteristics of these WGs, such as the graph error rate (GER) and the word density: the former is an indication of the compactness of the graphs, while the latter is a lower bound of achievable WER. Clearly, a stronger top-down knowledge (tri- instead of bi-grams) considerably reduces the graph size.

The **second layer** is sketched in Fig. 1 and is the cascade of two stages, WG expansion and rescoring, and decoding. The rescoring/expansion is obtained by matching every available example, among the 2.8 millions present in the template dataset, against every candidate triphone on the WG. For each comparison, 1) a Dynamic Time Warping (DTW) score has to be computed, 2) best-matching templates have to be sorted by least distance, and 3) triphones are expanded 50 times (i.e. 50 new arcs are created) and best DTW scores are assigned to the arcs.

A good graph is crucial for a template system like ours, more than it is for HMM-based speech recognizers: the phone graph must be as small as possible to reduce the computation time for the WG recoring, which is very expensive. It is important to stress that the phone graph must also contain paths corresponding to the true transcription because the template matching will not emit any new hypothesis: the only purpose of the template matching is to re-estimate arcs' acoustic weights in the light of the new template paradigm. Finally, a Viterbi search (with open beam) is run on the newly expanded WG. As it might be expected also this decoding phase is much more expensive than running the search on the original graph.

4. Pruning

To prune wrong or unnecessary examples from the large template database, we resorted to an iterative evaluation of the training material. The idea is to eliminate, at each iteration, those templates that are potentially responsible for recognition errors. Therefore, each iteration can be outlined as the succession of the following three steps:

pr. 1 Decode the WGs; evaluation (WER).

pr. 2 Select the templates aligned to wrongly recognized words.

pr. 3 **a)** Prune all selected templates.

OR

b) Compute the occurrence histogram of the templates selected at **pr. 2**; prune only those templates, which belong to the tail of the histogram (one occurrence only);

No retraining is necessary at all, as the model *is* the data: such an approach would have been practically unfeasible using a large parametric model, such as large HMMs, because even few iterations would have taken months with the current technology.

The selection step (**pr. 2**) can be optionally reinforced by an inspection of the histogram of the selected templates (**pr. 3b**): instead of deleting all templates (**pr. 3a**) we can alternatively proceed more conservatively and eliminate singletons only, i.e. the templates that appear only once at each iteration and that

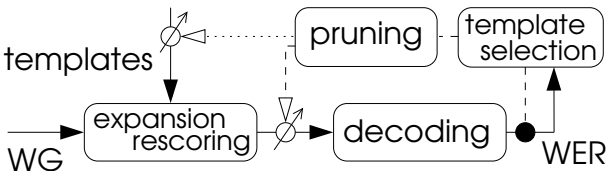

Figure 1: *Second layer: WG rescoring/expansion using templates, and WG decoding (and evaluation). The final WER can be used as feedback information to either prune the rescored WG (dashed line), or the template database (dotted line).*

Table 2: *Baseline: On the left (***a***), Word densities and Graph Error Rate (GER) of the WGs obtained after the first decoding layer. Next, the performance of WG decoding using (HMM) acoustic models, in the middle (***b***), and using templates, on the right (***c***). Performance is measured in Word Error Rate (WER). Tests on the* WSJ *were done using an extended bi-gram LM to cope with OOV words.*

	Word density			GER			WER			WER		
LM	WSJ	dev.92	eva.92	WSJ	dev.92	eva.92	WSJ	dev.92	eva.92	WSJ	dev.92	eva.92
2-gr.	1.9	7.40	7.13	0.26	2.53	2.68	1.32	10.17	9.78	2.47	11.97	12.01
3-gr.	-	5.32	5.16	-	2.54	2.68	-	7.99	8.17	-	9.32	9.50

are confined in the tail of the error distribution. The rationale behind this approach is that, due to the large number of templates in the database, the chance of selecting several bad templates per iteration is low; the mode of the histogram is more likely due to good, frequent, more prototypical templates which are nevertheless employed for the recognition of—often only partially—wrong words. The effect of the pruning using the information in the histogram can be appreciated by comparing Fig. 3 and 4 and will be discussed in the next section.

Practically, such a feedback algorithm can be applied at two different levels in our system. Templates can be eliminated from the template dataset, i.e. WG rescoring should be repeated at each iteration (this is represented by the dotted line in Fig. 1); alternatively they can be pruned from the WGs we are optimizing the pruning on (dashed line in Fig. 1); in this latter case only WG decoding must take place at each iteration. The former approach is possibly more precise because the WG arcs are expanded with a relatively small number of templates to keep the size reasonably limited, however this certainly implies higher computation costs. For this reason, for the experiments presented in this paper we opted for the latter approach, which we deemed a fair approximation of database pruning. Note that both approaches can be combined, and that, for instance, the WGs can be rescored again with the surviving templates in the database after a fixed number of decoding and selection steps.

Preliminary results showed that the template selection based on the dev.92 set is not sufficiently robust because of lack of acoustic variability: the dev.92 data contain too few speakers (10). For this reason, we optimized the pruning iterations on the template dataset directly by cross validation. More specifically, we resorted to leave-one-speaker-out: the WSJ training dataset was split into 284 parts, each part consisting of all the utterances—one WG per utterance—from the same speaker; the template-based ASR was then tested 284 times on each speaker individually exploiting the templates from all other speakers. Additionally, OOV words were added to the bi-gram LM used for optimization: in this way we avoided to select those words that could have merely been OOV. The starting WER over all speakers was quite low 2.47% and could be reduced to 0.26% (cf. Tab. 2.c) during the optimization.

5. Experiments and results

The iterative pruning algorithm explained in Sec. 4 was run for a number of times until a minimum in the WER was found. The optimization using the algorithm without histogram pruning (**pr. 3a**) is shown in Fig. 3: a minimum is found relatively soon, at 12 iterations. However, the list of templates to be discarded at this point is already rather populated, corresponding to approximately 11% of the entire template dataset. In Fig. 2 we tested the pruned template database on the eva.92 evaluation set, using bi- or tri-grams. The WER remained unaltered, while the size of the WGs was hugely reduced, approximately by 12%. It is worth mentioning that the pruning is equally effective on WGs in combination with tri-grams, which are generally smaller and of better quality in terms of WER.

Fig. 4 shows the optimization curve of pruning when the templates to be eliminated belong to the tail of their distribution (**pr. 3b** in Sec. 4). The minimum is reached after a larger number of iterations (39) because the selection of bad templates is more selective and more precise: the WER in the minimum is 50% lower than if all templates were used. Although the number of pruned templates also grows slowly, a much larger number of examples are eventually marked for deletion. More specifically the pruned database is now almost 29% smaller and the WGs of the eva.92 dataset (see Fig. 2) are 30% smaller independently of the LM used. Using the complete pruning algorithm, however, does not improve the WER on the evaluation set either. These results corroborate our assumption that wrong words contain many good, though redundant, templates too.

The robustness and the stability of the pruning can be ascertained from the boxes at the bottom of Fig. 3 and 4. A number ($>$30) of random pruning steps was also performed on the training set and the respective pruned template databases were tested on the eva.92 set as well. In this way we were able to compare the algorithm described in Sec. 4 with a pure random selection. The average results and the relative standard errors of random pruning are shown in gray in Fig. 4; the results are always significantly worse than the complete pruning algorithm. This is not the case if histogram information is not used (**pr. 3a**). From the same figures we can appreciate how stable the pruning is w.r.t. the number of iterations. The hull of the minimum is wide during optimization and almost flat in the evaluation phase.

6. Conclusions and future work

In this work we presented a framework characterized by the symbiosis between data pruning and template-based ASR. Pruning by WER optimization was feasible only thanks to the non-parametric nature of the system. On the other side, pruning

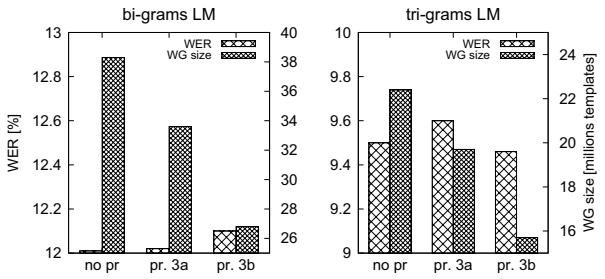

Figure 2: *Pruning effects in terms of WG size (# of arcs) and WER (%) obtained on the* eva.92 *evaluation set. Results without (***pr. 3a***) and with (***pr. 3b***) histogram pruning. Bi- and tri-gram language models are also compared.*

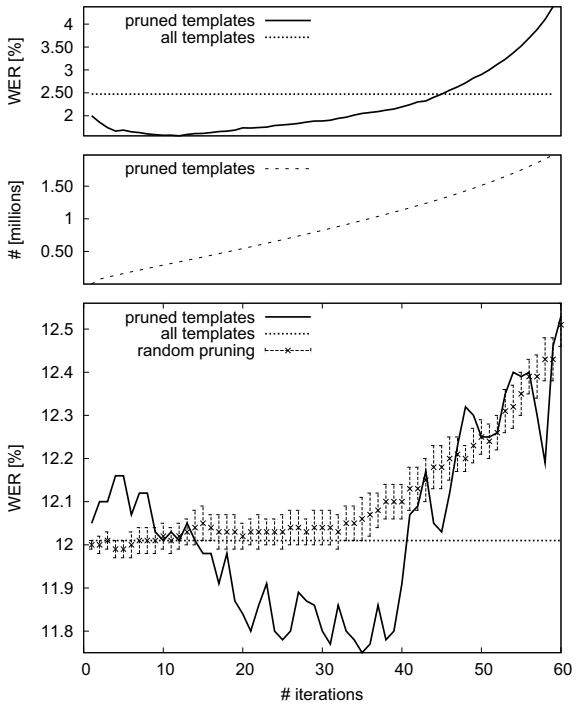

Figure 3: *Example of template pruning with* bi-grams. *Above: optimization* without *histogram pruning (***pr. 3a***). Middle: number of templates pruned at each iteration. Below: pruning evaluation on the* eva.92 *dataset compared to random, multiple pruning (mean and standard errors are shown).*

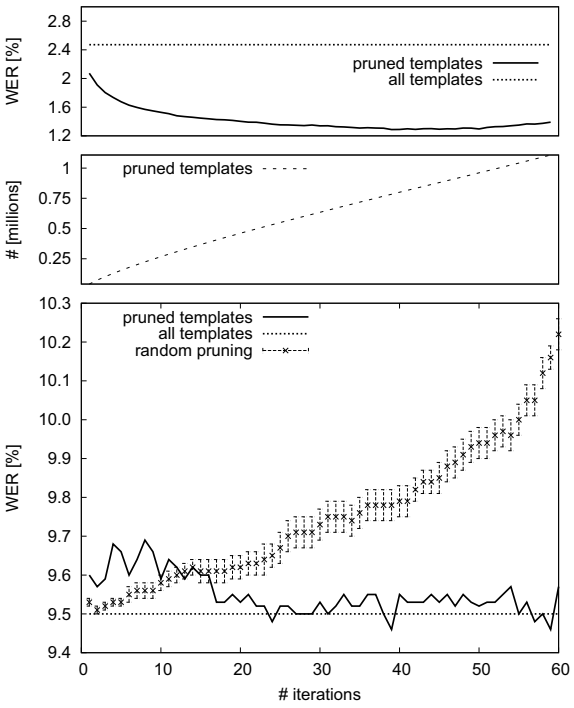

Figure 4: *As Fig. 3, with* tri-grams. *Template pruning by exploiting histogram information (***pr. 3b***).*

was impressively effective in reducing the 'model' size of the T-ASR: the template dataset could be reduced by almost 30% relative, and WG size decreased by quite the same amount.

However, further inspection of the results suggests that even more can be probably gained: although the choice of the objective function would have suggested differently, the WER did not decrease. This needs further investigation because a deep minimum of the WER has been observed during the pruning of the train database; the drop in the WER, however, does not generalize to unseen data and the minimum (if present) found on the evaluation data is shifted (Fig. 3). In the future, we plan to tackle this problem and look for accuracy improvements.

We would like to stress the fact that a much more compact set of templates does not only lead to a huge reduction of memory usage and of computation time for rescoring and decoding. It also paves the road to more elaborate search algorithms and to the exploitation of additional databases. So far, complex searches that consist of, e.g., introducing additional costs on the WG arcs, have been too time consuming but can now be speeded up considerably.

Another valuable feature of the template selection proposed in this paper is the analytical framework it offers. Templates that are responsible of recognition errors can be easily identified and subsequently analyzed. In this way, we hope that our system will help shedding more light on the contribution of phonetic details to ASR.

Finally, the pruning itself can be ameliorated: pruning involves hard decisions; a template is either eliminated or not. This could be detrimental as too many examples can be discarded too early: in general, decisions should be postponed as much as possible (principle of *least commitment*). By setting a penalizing score at WG arcs instead of deleting them, we could probably obtain even more reliable WGs. Hard decisions have the advantage of not introducing additional weights and of reducing the size of the graphs. However, further analyses are needed of penalizing scores that rely on the same framework as described in this paper.

7. References

[1] F. Jelinek, "Some of my best friends are linguists," in *Proc. of LREC*, Lisbon, Portugal, 2004.

[2] L. Lamel, J.L. Gauvain, and G. Adda, "Lightly supervised and unsupervised acoustic model training," *Computer Speech and Language*, vol. 16, no. 1, pp. 115–229, 2002.

[3] Y. Wu, R. Zhang, and A. Rudnicky, "Data selection for speech recognition," in *Proc. of ASRU*, Kyoto, Japan, 2007, pp. 562–565.

[4] A. Angelova, Y. Abu-Mostafa, and P. Perona, "Pruning training sets for learning of object categories," in *Proc. of CVPR*, San Diego, USA, 2005, pp. 494–501.

[5] N. Matic, I. Guyon, L. Bottou, J. Denker, and V. Vapnik, "Computer aided cleaning of large databases for character recognition," in *Proc. of ICPR*, The Hague, The Netherlands, 1992, pp. 330–333.

[6] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, 2003.

[7] S. Demange and D. Van Compernolle, "HEAR: An hybrid episodic-abstract speech recognizer," in *Proc. of INTERSPEECH*, Birmingham, UK, 2009, pp. 3067–3070.

[8] M. De Wachter, M. Matton, K. Demuynck, P. Wambacq, R. Cools, and D. Van Compernolle, "Template-Based Continuous Speech Recognition," *IEEE Transactions on ASLP*, vol. 15, no. 4, pp. 1377–1390, 2007.

[9] K. Demuynck, J. Roelens, D. Van Compernolle, and P. Wambacq, "SPRAAK: an open source 'SPeech Recognition and Automatic Annotation Kit'," in *INTERSPEECH*, Brisbane, 2008, p. 495.