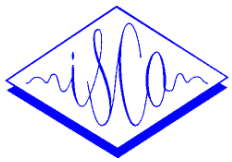

6th International Conference on Spoken Language Processing (ICSLP 2000)
Beijing, China
October 16-20, 2000

PARTICLE-BASED LANGUAGE MODELLING

*E.W.D.Whittaker** *P.C.Woodland*

Cambridge University Engineering Department,
Trumpington Street, Cambridge CB2 1PZ, UK.
{ewdw2,pcw}@eng.cam.ac.uk

ABSTRACT

This paper investigates the use of particle (sub-word) N-grams for language modelling. One linguistics-based and two data-driven algorithms are presented and evaluated in terms of perplexity for Russian and English. Interpolating word trigram and particle 6-gram models gives up to a 7.5% perplexity reduction over the baseline word trigram model for Russian. Lattice rescoring experiments are also performed on 1997 DARPA Hub4 evaluation lattices where the interpolated model gives a 0.4% absolute reduction in word error rate over the baseline word trigram model.

1. INTRODUCTION

Most of the current approaches to language modelling for speech recognition tend to use words, or classes of words, as the modelling units. Words are a logical choice, since it is ultimately words that are to be output by a speech recognition system, but they are not necessarily the best units for capturing dependencies in a text. The optimal set of units will inevitably depend on the language, the sparsity of the training data and the model framework in which the units are used. This work was motivated in part by the problems encountered in language modelling of Russian. In [7] the need for an alternative approach to word-based language modelling of Russian was highlighted. In particular, the very large vocabulary size that was required to achieve a usable OOV-rate with Russian (430k words for a 1.1% OOV-rate) indicated that modelling at the word level was possibly not the best solution. Russian words have many morphological units in common and this is particularly evident in the case of Russian word inflections, which are appended to word stems to denote the word's grammatical case, gender and number. Given this characteristic of the Russian language, there is a strong argument for decomposing words into units, smaller than the word themselves, and using these in some form of language model. The focus of the work presented here is the selection of sub-word units (referred to here as particles) for use in N-gram language models. The particle model is introduced in the next section and the three algorithms are described in Section 3 together with the results of perplexity experiments. Lattice rescoring experiments are described in Section 4 and followed by conclusions and further work.

2. THE PARTICLE N-GRAM MODEL

In contrast to similar approaches in which alternative modelling units to words have been investigated [3, 5] the method adopted here enforces a deterministic decomposition of a given word into particles and does not permit particles to span word boundaries. This paper investigates several methods for the automatic generation of a deterministic decomposition function U between words and a set Ψ of particles u_i

$$U : w \rightarrow U(w) = u_0, u_1, \ldots, u_{L(w)-1} \quad u_i \in \Psi.$$

where word w is decomposed into a sequence of $L(w)$ particles. In this work a string-matching algorithm is evaluated together with two greedy data-driven algorithms, the objectives of which are to determine the particle units and word decompositions that best model the training data.

Identification of word boundaries at the particle level is necessary to ensure a deterministic mapping from a sequence of particles back to the word-level. Therefore, a <w> symbol is always attached to the terminal particle $u_{L(w)-1}$ in the decomposition of word w to denote a word boundary. Otherwise, a summation over all the possible ways in which the word stream could be obtained from the particle stream would be necessary. This is generally undesirable when incorporating the language model into a speech recogniser. A consequence of this requirement is that the relationship between some morphological strings is inevitably lost. For example, for the two words "SATISFY" and "SATISFYING", if "SATISFY", "SATISFY<w>" and "ING<w>" are determined to be particles, the connection between the two words is lost. However, from the point of view of language modelling, the inclusion of the word boundary marker may be seen as incorporating additional linguistic information. Strings which appear at the ends of words cannot be confused with identical strings which appear in other word positions.

Once the decomposition function U has been determined, words in the training text can be decomposed and the probability of some text computed using a particle N-gram language model:

$$P(w_1, \ldots, w_n) \approx \prod_{i=1}^{N_P} P(u_i \mid u_{i-N+1}, \ldots, u_{i-1}), \quad (1)$$

where each word w in the text is decomposed using $U(w)$ into its constituent particles and N_P is the total number of particles in the text when all words have been decomposed. Relative frequencies of the occurrences of particle N-tuples can be used to compute the above conditional particle probabilities and then smoothed in a manner similar to that used for conditional word probabilities. It is assumed here that it is ultimately words which are to be recognised and therefore the discussion here only concerns word level probabilities and perplexities. It must also be emphasised that the particle units under discussion are not being proposed as sub-word units for acoustic modelling, but only as units for language

*E.W.D.Whittaker was supported in this research by an EPSRC studentship. His current affiliation is with Compaq Computer Corporation's Cambridge Research Laboratory (edw@crl.dec.com).

modelling. The word-level probability of a word w_n given some particle context $h = U(w_1) \ldots U(w_{n-1})$ can be computed using the following particle bigram model for example:

$$P(w_n \mid h) = \frac{1}{Z(h)} \cdot P(u^{w_n}_{L(w_n)-1} \mid u^{w_n}_{L(w_n)-2}) \cdot P(u^{w_n}_{L(w_n)-2} \mid u^{w_n}_{L(w_n)-3}) \cdots P(u^{w_n}_0 \mid u^{w_{n-1}}_{L(w_{n-1})-1}), \quad (2)$$

where w_n is decomposed into $L(w_n)$ individual particles $u^{w_n}_i$ for $i = 0, \ldots, L(w_n) - 1$, and w_{n-1} is decomposed into $L(w_{n-1})$ individual particles $u^{w_{n-1}}_j$ for $j = 0, \ldots, L(w_{n-1}) - 1$, and $Z(h)$ is a normalization constant. In particular, the bigram context for the first particle of word w_n (which is identical to h in the above example) is the terminal particle of the previous word: $u^{w_{n-1}}_{L(w_{n-1})-1}$. The normalization constant $Z(h) = \sum_{w_i} P(w_i \mid h)$ ensures a correct probability distribution at the word level. The remainder of the probability mass $(1 - Z(h))$ is accounted for by words that are not in the vocabulary, but for which probabilities could be generated from unused sequences of particles up to some maximum number of particles. This may be seen as a beneficial consequence of the particle modelling approach which permits the word-level vocabulary to be augmented with new words without the complete retraining of the particle language model. If all single letters and all single letters attached to the word boundary symbol were included in Ψ then a probability could be generated for all possible words.

3. PERPLEXITY EXPERIMENTS

In this section, we describe three automatic methods for decomposing words into particle units: a string-matching, *affix-stripping* algorithm, and two greedy, data-driven algorithms for optimising *particle selection* and *word decompositions* by maximising the likelihood of the training data using a particle bigram model. Each of the three algorithms is applied to 1) a 430k Russian vocabulary using a specially collected Russian corpus [7], and 2) a 65k English vocabulary using the British National Corpus [1]. Both corpora (each containing around 100 million words) were divided into `training`, `dev-test` and `eval-test` sets in the ratios 98:1:1 respectively. The latter two sets were used for parameter optimisation and model evaluation respectively. Each vocabulary (most frequent 430k Russian words and 65k English words in each `training` set) has a 1.1% OOV-rate on the corresponding `dev-test` and `eval-test` sets.

Each algorithm produces a set of decompositions for every vocabulary word which is then used to build particle 6-gram models on the `training` portion of each corpus. 6-grams were chosen for the particle models since on average there were around two particles per vocabulary word after decomposition and comparisons were to be made against word trigram models. The particle models were pruned using entropy-based pruning [6] so as to contain approximately 12 million parameters—approximately the same number as in the baseline word trigram model (pruned by discarding all singleton bigrams and trigrams) for each language. All models employ Katz backoff with Good-Turing discounting. The perplexity and number of parameters in the baseline word trigram model for each language are shown in Table 1.

For the purposes of comparison and to demonstrate the action

	Word trigram perplexity		Model size
Language	`training`	`eval-test`	(parameters)
Russian (430k)	463.5	677.0	12,177,700
English (65k)	162.5	216.1	12,431,060

Table 1: Perplexities of word trigram models on `eval-test` and `training` data partitions for the two corpora together with the number of parameters in each trigram model.

of the data-driven algorithms, particle bigram models were also built for each set of word decompositions. These models retained all bigram events and every token in the `eval-test` set was predicted in computing the particle bigram perplexity PP^{2g}_{part}. Due to the computational burden of computing the normalisation constant $Z(h)$ in Equation (2), this factor was omitted from all perplexity calculations. As a consequence, the perplexity figures of particle models represent upper bounds on their true values. The difference in perplexity was found to be no greater than 2.5% for stand-alone models. The final column of each table of results below, shows the improvement of the interpolated word trigram and particle 6-gram model over the stand-alone word trigram.

3.1. Affix-stripping algorithm

No satisfactory rule-based method was found for producing linguistically accurate decompositions of arbitrary Russian words so the first set of particle experiments was performed using a simple, string-matching algorithm to produce decompositions for the 430k Russian vocabulary and the 65k English vocabularies. For the Russian vocabulary, 28 prefixes and 60 suffixes were first chosen according to their usefulness and productivity (as perceived by the authors) with the help of a textbook of Modern Russian Grammar [4]. Similarly for the English vocabulary, 52 prefixes and 62 suffixes were obtained with the help of an encyclopedia of the English language [2]. The longest prefix and suffix, found using a simple string-matching operation, were then systematically separated from the beginning and end, respectively, of each vocabulary word. All vocabulary words were eligible for decomposition and the affixes were separated wherever a match was made, irrespective of whether the match was linguistically correct or not. The <w> symbol was then appended to the terminal particle (the suffix or the end of a word) in the decomposition for each word. The perplexities of the affixes models using the resulting word decompositions are shown in Table 2 for both languages.

	`training`		`eval-test`		%
Language	PP^{2g}_{part}	PP^{6g}_{part}	PP^{6g}_{part}	PP^{6g}_{int}	improv
Russian	1019	510.0	747.5	632.3	6.6
English	335.8	161.8	225.6	204.4	5.4

Table 2: Word level perplexities of stand-alone particle models on `training` and `eval-test`, and interpolated particle and word trigram model on `eval-test` only.

3.2. Particle selection algorithm

The particle selection algorithm uses only the word unigram and bigram statistics from the training data and a list of all possible candidate particles of different lengths. This list only contains those particles which actually occur *within* words of the vocab-

ulary. Initialising the algorithm involves decomposing all words into their constituent single characters. The contents of the set of particles Ψ at initialisation therefore comprise all single characters which occur in words of the vocabulary. Single characters must always appear in the final set since they may be necessary as *filler* particles to complete a decomposition which does not divide exactly into larger particles. The algorithm is described concisely by the following steps:

1. **Initialisation**:
   - $l = 1$
   - `decompose words into` l`-character particles`
   - `compute likelihood of training data`
2. $l = l + 1$
3. **Iterate** $\forall$ l`-character candidate particles` u^{can}`:`
   - `'insert' particle` u^{can} `in all words` w
   - `compute change in training set likelihood`
   - `'remove' particle` u^{can} `from all words` w
4. `Insert best` l`-character particle into` Ψ `and permanently in all words`
5. `If desired number of particles obtained then` **terminate**
6. `If no particles remaining then` **terminate**
7. `If improvement goto` **step 3**, `else goto` **step 2**

Each iteration involves a search over a set of particles of a fixed length l characters, at the end of which the particle that gave the greatest reduction in perplexity is permanently added to the final set of particles. The order in which particles are chosen affects the selection of all subsequent particles. Since the algorithm only accepts configurations which result in an increase in the optimisation function, the algorithm is guaranteed to converge, however due to its greedy nature it is only likely to find a locally optimal solution. In these experiments the algorithm is only used to determine a set of particles up to some maximum size l_{max} and does not run to completion. The perplexities of models built using decompositions for different values of $l_{max} = 1 \ldots 5$ are shown in Table 3 for Russian and Table 4 for English.

	training		eval-test		%
l_{max}	PP^{2g}_{part}	PP^{6g}_{part}	PP^{6g}_{part}	PP^{6g}_{int}	improv
1	347300	1529	1784	662.9	2.1
2	25910	695.2	897.8	630.6	6.9
3	4171	591.8	800.2	626.2	7.5
4	1575	542.6	766.0	627.1	7.4
5	979.2	515.4	750.0	630.1	6.9

Table 3: Russian corpus (430k): word level perplexities of stand-alone particle models on `training` and `eval-test`, and interpolated particle and word trigram model on `eval-test` only.

	training		eval-test		%
l_{max}	PP^{2g}_{part}	PP^{6g}_{part}	PP^{6g}_{part}	PP^{6g}_{int}	improv
1	93900	429.4	472.3	214.4	1.7
2	6127	217.1	272.4	209.0	3.3
3	1040	191.8	250.3	206.2	4.6
4	462.2	175.2	238.4	204.5	5.4
5	315.3	166.7	231.7	203.9	5.6

Table 4: English corpus (65k): word level perplexities of stand-alone particle models on `training` and `eval-test`, and interpolated particle and word trigram model on `eval-test` only.

3.3. Word decomposition algorithm

The word decomposition algorithm assumes an initial set of word decompositions and then iteratively optimises the decompositions for each word in turn. For the experiments here different initialisations were generated by (i) collecting all word-internal particle bigram statistics for all particles up to some maximum number of characters l_{max}, then (ii) determining the highest probability decomposition of each vocabulary word. The output of the affix-stripping algorithm was also investigated as an initialisation. The data-driven initialisation procedure and the algorithm itself are described concisely by the following steps:

1. **Initialisation:**
   - `Limit maximum particle length to` l_{max} `characters`
   - `Collect word-internal particle statistics`
   - `Determine highest probability decomposition for each word and collect particle bigram statistics`
2. **Iterate** $\forall$ `words` w`:`
   - **Iterate** $\forall$ `word decompositions` $U(w)$`:`
     - `compute training set likelihood using current decomp. statistics`
   - `Select best word decomposition`
   - `Update particle statistics`
3. `Repeat` **step 2** `for fixed number of iterations`

The perplexities of models using the initialisation decompositions (*ini*) and the optimised (*opt*) decompositions obtained after one iteration through the vocabulary are given in Table 5 for Russian and Table 6 for English. Results using the decompositions of the affix-stripping algorithm after application of the word decomposition algorithm are also shown.

	training		eval-test		%
Model	PP^{2g}_{part}	PP^{6g}_{part}	PP^{6g}_{part}	PP^{6g}_{int}	improv
2 (ini)	21110	706.9	905.8	635.1	6.2
2 (opt)	13160	678.7	876.0	630.3	6.9
3 (ini)	2941	611.4	821.4	635.8	6.1
3 (opt)	1573	572.6	779.4	628.2	7.2
4 (ini)	1423	589.9	813.6	641.1	5.3
4 (opt)	729.2	532.1	769.6	635.6	6.1
affixes(opt)	905.0	518.8	751.0	633.3	6.5

Table 5: Russian corpus (430k): word level perplexities of stand-alone particle models on `training` and `eval-test`, and interpolated particle and word trigram model on `eval-test` only.

	training		eval-test		%
Model	PP^{2g}_{part}	PP^{6g}_{part}	PP^{6g}_{part}	PP^{6g}_{int}	improv
4 (ini)	458.2	185.0	245.4	206.7	4.3
4 (opt)	293.2	174.4	237.2	205.2	5.0
5 (ini)	398.5	177.4	240.4	206.5	4.4
5 (opt)	252.8	166.1	230.8	204.9	5.2
6 (ini)	378.9	178.2	242.9	207.0	4.2
6 (opt)	243.1	166.6	232.8	205.4	5.0
affixes(opt)	310.3	167.2	227.2	205.3	5.0

Table 6: English corpus (65k): word level perplexities of stand-alone particle models on `training` and `eval-test`, and interpolated particle and word trigram model on `eval-test` only.

3.4. Discussion

The results for the interpolated word and particle models show similar improvements over the baseline word trigram for each of the three algorithms that were investigated. The improvements were greater for Russian than for English and interestingly the two data-driven algorithms showed the greatest improvement for $l_{max} = 3$. An examination of the particles obtained with these models showed many of them to be recognisable morphological units. For English, as l_{max} was increased for the particle selection algorithm, the perplexity decreased. This was found to be due to the inclusion of ever more whole words in the set of particles. In the 6-gram framework these were generally beneficial. However, the results of the word decomposition algorithm on English showed comparatively little variation in perplexity with different l_{max} and this was attributed to the intialisation method which did not favour the selection of whole words. For both algorithms the bigram optimisation criterion generalised reasonably well to the final 6-gram models. The only exception was initialising the word decomposition algorithm with the decompositions from the affix-stripping algorithm which produced worse decompositions for the 6-gram model on both data sets.

4. RECOGNITION EXPERIMENTS

The recognition performance of the particle models was evaluated by rescoring word trigram lattices that had been generated using the 1997 HTK broadcast news transcription system [8] (unfortunately a Russian recognition system was not available). The language model training data comprised 132 million words of the LDC broadcast news texts, the transcriptions of the 1997 broadcast news training data (added twice) and the 1995 Marketplace transcriptions. A word trigram model was built using the same vocabulary that was used to generate the original lattices. This baseline word trigram employed Katz backoff with Good-Turing discounting and had singleton bigrams and singleton and doubleton trigrams removed to give a model containing around 12 million parameters. A set of word decompositions was generated using each of the three algorithms ($l_{max} = 5$ for the two data-driven algorithms) and particle 6-gram models were built in an identical manner to that described in the previous section.

Model	Modelling units	PP_{eval}	$\%WER$
Affix-stripping	47,105	191.5	18.6
Particle Selection	17,061	188.0	18.5
Word Decomposition	34,991	190.0	18.4
Word trigram	65,425	185.7	18.1

Table 7: Results of stand-alone particle and word models on Hub4 eval97 data together with number of modelling units used in each model.

Interpolated model	PP_{eval}	$\%WER$	% rel. imp.
Affix-stripping	174.5	17.8	1.7
Particle Selection	171.8	17.9	1.1
Word Decomposition	173.2	17.7	2.2

Table 8: Results of interpolated word & particle models on Hub4 eval97.

The word error rate ($\%WER$) results on the 1997 Hub4 evaluation lattices are given in Table 7 for the stand-alone particle models and the baseline word trigram. The perplexity of each model on the reference transcriptions is also given. The performance difference between the stand-alone word and particle models is relatively small even though far fewer modelling units were used in the particle models. In Table 8 perplexity, $\%WER$ and relative improvement in $\%WER$ over the baseline word trigram are given for the interpolated word and particle models. The interpolation weights were optimised using the 1997 Hub4 development lattices and set to 0.6 and 0.4 for the word and particle models respectively. The interpolated models give only a small reduction in $\%WER$ despite the relatively large reduction in perplexity. The smoothing effect of the particle model is limited for this particular data since the word model was already well trained. The perplexity reductions are due in part to the boosting of already well estimated events and the inclusion of some higher-order word N-grams.

5. CONCLUSION

This paper has investigated a language modelling approach which uses particles as the modelling units and was motivated by the productive morphology of Russian. By incorporating particles into the N-gram framework, stand-alone particle models gave similar perplexity and $\%WER$ results to the word model despite capturing completely different language dependencies. In addition, reductions in perplexity were obtained for interpolated word and particle models over either of the individual models. Small reductions in word error rate were also obtained on a broadcast news task using interpolated word and particle models.

The experiments presented here form the basis for further work to investigate the incorporation of longer-range dependencies between particles. Such linguistic dependencies are known to be particularly strong in languages like Russian and their inclusion would undoubtedly complement the somewhat restricted span of the particle N-gram model. It would also be interesting to assess the impact of the particle model in a recognition system for Russian.

6. REFERENCES

1. L. Burnard. *Users Reference Guide for the British National Corpus*. Oxford University Computing Services, 1995.
2. D. Crystal. *The Cambridge Encyclopedia of the English Language*. Cambridge University Press, 1995.
3. S. Deligne and F. Bimbot. Inference of Variable-length Linguistic and Acoustic Units by Multigrams. *Speech Communication*, 23:223–241, 1997.
4. D. Offord. *Modern Russian: An Advanced Grammar Course*. Bristol Classical Press, 1993.
5. K. Ries, F. Dag Buo, and A. Waibel. Class Phrase Models for Language Modelling. In *Proceedings of ICSLP'96*, Philadelphia, USA, 1996.
6. A. Stolcke. Entropy-based Pruning of Backoff Language Models. In *Proceedings of 1998 DARPA Broadcast News Transcription and Understanding Workshop*, 1998.
7. E. Whittaker and P. Woodland. Comparison of Language Modelling Techniques for Russian and English. In *Proceedings of ICSLP'98*, Sydney, Australia., 1998.
8. P. Woodland, T. Hain, S. Johnson, T. Niesler, A. Tuerk, E. Whittaker, and S. Young. The 1997 HTK Broadcast News Transcription System. In *Proceedings of 1998 DARPA Broadcast News Transcription and Understanding Workshop*, 1998.