Focusing Language Models For Automatic Speech Recognition

Daniele Falavigna, Roberto Gretter

HLT research unit, FBK, 38123 Povo (TN), Italy

(falavi,gretter)@fbk.eu

Abstract

This paper describes a method for selecting text data from a corpus with the aim of training auxiliary Language Models (LMs) for an Automatic Speech Recognition (ASR) system. A novel similarity score function is proposed, which allows to score each document belonging to the corpus in order to select those with the highest scores for training auxiliary LMs which are linearly interpolated with the baseline one. The similarity score function makes use of "similarity models" built from the automatic transcriptions furnished by earlier stages of the ASR system, while the documents selected for training auxiliary LMs are drawn from the same set of data used to train the baseline LM used in the ASR system. In this way, the resulting interpolated LMs are "focused" towards the output of the recognizer itself.

The approach allows to improve word error rate, measured on a task of spontaneous speech, of about 3% relative. It is important to note that a similar improvement has been obtained using an "in-domain" set of texts data not contained in the sources used to train the baseline LM.

In addition, we compared the proposed similarity score function with two other ones based on perplexity (PP) and on TFxIDF (Term Frequency x Inverse Document Frequency) vector space model. The proposed approach provides about the same performance as that based on TFxIDF model but requires both lower computation and occupation memory.

1. Introduction

Automatic speech recognition systems can significantly take advantage from training language models on large text corpora that represent well the application domain. Since, generally, a limited amount of in-domain text data is available for a given application, from which a corresponding LM is trained, the acquisition of more domain specific text data often becomes a crucial task.

It is a common practise among ASR specialists to try to automatically obtain texts relevant for the given application from large publicly available corpora and to use the collected corpora to train auxiliary LMs to be combined with the in-domain LM.

In the literature several methods are proposed for selecting text data matching an in-domain LM. In general, the approaches consist in using a function that gives a similarity score to each possible candidate text (sentences or entire documents) to select and to retain only those whose scores are higher than a predefined threshold.

In [1] the similarity function used to score documents is simply the perplexity computed using the given in-domain LM.

The work reported in [2] utilizes two unigram LMs, both trained on the general corpus to select: the first LM is trained on all texts of the corpus, the second LM is trained on all texts except the document to score. The difference in the log-likelihood of the in-domain text data given by the two LMs is used as scoring function.

The work in [3] proposes a method based on cross-entropy difference between the in-domain LM and a LM trained on a random sample of the general text data to select. The authors of this paper demonstrated significant reduction of perplexity using this method, with respect to [1] and [2], on a corpus used for automatic Machine Translation (MT).

In [4] three data selection techniques are proposed. The first one is based on a vector space model that uses TFxIDF (Term Frequency x Inverse Document Frequency) feature coefficients. A centroid similarity measure, defined as scalar product between a vector representing in-domain data and a vector representing the document to score, is employed. The second and the third methods are based on an "ngram-ratio" similarity measure and on ranking the documents of the general text corpus through resampling of in-domain data, respectively. The paper shows improvements both in perplexities and BLEU scores using all of the three selection methods. In addition, the paper demonstrates that the automatic selection approaches work well even if the set of in-domain text data, on which similarity models are estimated (both LMs or TFxIDF vectors), is replaced by texts coming from the output of the MT decoder.

More recently, some approaches have been proposed for adapting LMs using data extracted from the Web. The authors of [5] compare the usage of both manually and automatically generated texts for selecting auxiliary data for LM adaptation in a ASR task. In [6] a strategy is proposed for automatic closed-captioning of video that uses a LM adapted to the topic of the video itself. A classification is first performed to determine the topic of a given video and a large set of topic-specific LMs is trained using documents downloaded from the Web.

This work has been partially funded by the European project EU-BRIDGE, under the contract FP7-287658.

Similarly to [4] and [5] we use automatically generated documents (i.e. the documents obtained from the automatic transcriptions of the audio) to select text data from a huge general text corpus. Given an automatically transcribed document (the query document), the purpose of the selection procedure is to detect and retain from the general corpus only the documents that are most similar to a given query. Then, an auxiliary LM is trained using the automatically (query dependent) selected data. However, differently from [4], [5] and [6] we select documents for training the auxiliary LMs from the same set used to train the baseline LM employed in the ASR system, i.e. no additional documents are required to train auxiliary LMs. Finally, baseline and auxiliary LMs are linearly interpolated, as will be explained below.

This procedure allows to train LMs focused on the query document, i.e. on the ASR output. We prefer to use the term "LM focusing", instead of LM adaptation, to underline the fact that we are not using new data to train auxiliary LMs but, on the contrary, a subset of existing text data is somehow enhanced in order to better match the linguistic content of the audio to transcribe. To be more precise, we are proposing to "frequently" adapt the LM according to a given (or automatically detected) segmentation of the audio stream to transcribe. Since to do this it is necessary to train auxiliary LMs through data selection over large corpora of text data we developed an approach, similar to to TFxIDF based one, that employs a vector space model to represent documents to compare. However, the employed features, the way adopted for storing them and the similarity metrics used, has allowed to improve both computation and memory efficiency with respect to TFxIDF. In section 3 the detailed description of the proposed method and comparisons with both TFxIDF method and an approach based on perplexity minimization will be given.

The source used for LMs training is "google-news", an aggregator of news, provided and operated by Google, that collects news from many different sources, in different languages, and that groups articles having similar contents. We download daily news from this site, filter-out unuseful tags and collect texts. Therefore, a "google-news" corpus has become available for training both baseline LM and auxiliary ones.

To measure the performance of our automatic selection approach we carried out a set of experiments on the evaluation sets delivered for IWSLT 2011 Evaluation Campaign[1]. Task of this campaign is the automatic transcription/translation of TED talks, a global set of conferences whose audio/video recordings are available through the Internet (see http://www.ted.com/talks).

The simplest way for combining LMs trained on different sources is to compute the probability of a word w, given its past history h, as:

$$P[w \mid h] = \sum_{j=1}^{j=J} \lambda_j P_j[w \mid h] \quad (1)$$

where $P_j[w \mid h]$ are LM probabilities trained on the j^{th} source, λ_j are weights estimated with the aim of minimizing the overall perplexity on a development set and J is the total number of LMs to combine. More complex approaches [7] are based on linear interpolation of log-probabilities using discriminative training of λ_j (a comparison among different LM combination techniques can be found in [8]).

According to what previously seen, equation 1 is used to combine two LMs: the baseline LM (LM_{base}) and an auxiliary, i^{th} "talk-specific" LM (LM^i_{aux}), trained on auxiliary data, automatically selected. In particular, a preliminary automatic transcription of the given i^{th} TED talk is used both to select the data to train LM^i_{aux} and to estimate interpolation weights, λ^i_{base} and λ^i_{aux}, to be used with equation 1. Then, a rescoring ASR step is carried out, as explained in section 2.4, using focused, talk-specific LM probabilities given by equation 1.

We measured on IWSLT 2011 evaluation sets a relative improvement of about 3% in Word Error Rate (WER) after ASR hypotheses rescoring using auxiliary LMs trained on data selected with the proposed approach. The same improvement has been measured using TFxIDF based method for selecting auxiliary texts but, as previously mentioned, the latter method is more expensive both in terms of computation and memory requirements. Finally, a relative lower WER improvement has been achieved using an automatic selection procedure based on perplexity minimization.

2. Automatic Transcription System

The automatic transcription system used in this work is the one described in [9, 10]. It is based on two decoding passes followed by a third linguistic rescoring step.

For IWSLT 2011 evaluation campaign speech segments to transcribe have been manually detected and labelled in terms of speaker names. Then, audio recordings with manual segments to transcribe have been furnished to participants, hence no automatic speaker diarization procedure has been applied.

In both first and second decoding passes the system uses continuous density Hidden Markov Models (HMMs) and a static network embedding the probabilities of the baseline LM. A frame synchronous Viterbi beam-search is used to find the most likely word sequence corresponding to each speech segment to recognize. In addition, in the second decoding pass the system generates for each speech segment a word graph (see below for the details). The best word sequences generated in the second decoding pass are used to evaluate the baseline performance, as well as for selecting auxiliary documents. The corresponding word graphs are rescored in the third decoding pass using the focused LMs. Note that in this latter decoding step acoustic model probabil-

[1] visit http://www.iwslt2011.org/ for details of the IWSLT 2011 evaluation campaign

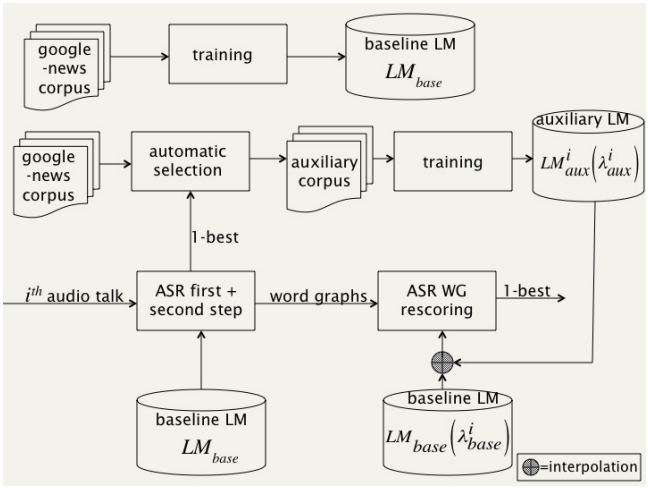

Figure 1: *Block diagram of the ASR system.*

ities associated to arcs of word graphs remain unchanged, i.e. the third decoding step implements a pure linguistic rescoring. Figure 1 shows a block diagram of the ASR system with the main modules involved, emphasizing both the procedure for selecting auxiliary documents and the rescoring pass using interpolated LM probabilities given by equation 1. More details related to each module are reported below.

2.1. Acoustic data selection for training

For acoustic model (AM) training, domain specific acoustic data were exploited. Recordings of TED talks released before the cut-off date, 31 December 2010, were downloaded with the corresponding subtitles which are content-only transcriptions of the speech. In content-only transcriptions anything irrelevant to the content is ignored, including most non-verbal sounds, false starts, repetitions, incomplete or revised sentences and superfluous speech by the speaker. The collected data consisted in 820 talks, for a total duration of ≈216 hours, with ≈166 hours of actual speech. The provided subtitles are not a verbatim transcription of the speeches, hence a lightly supervised training procedure was applied to extract segments that can be deemed reliable. The approach is that of selecting only those portion in which the human transcription and an automatic transcription agree (see [9, 11] for the details). This procedure has allowed to make available 87% of the training speech, and this amount was considered satisfactory.

2.2. Acoustic model

13 Mel-frequency cepstral coefficients, including the zero order coefficient, are computed every 10ms using a Hamming window of 20ms length. First, second and third order time derivatives are computed, after segment-based cepstral mean subtraction, to form 52-dimensional feature vectors. Acoustic features are normalized and HLDA projected to obtain 39-dimensional feature vectors as described below.

AMs were trained exploiting a variant of the speaker adaptive training method based on Constrained Maximum Likelihood Linear Regression (CMLLR) [12]. In our training variant [13, 10] there are two sets of AMs, the target models and the recognition models. The training procedure makes use of an affine transformation to normalize acoustic features on a cluster by cluster basis (a cluster contains all of the speech segments belonging to a same speaker, according to the given manual segmentation) with respect to the target models. For each cluster of speech segments, an affine transformation is estimated through CMLLR [12] with the aim of minimizing the mismatch between the cluster data and the target models. Once estimated, the affine transformation is applied to cluster data. Recognition models are then trained on normalized data. Leveraging on the possibility that the structure of the target and recognition models can be determined independently, a Gaussian Mixture Model (GMM) can be adopted as target model for training AMs used in the first decoding pass [13]. This has the advantage that, at recognition time, word transcriptions of test utterances are not required for estimating feature transformations. Instead, target models for training recognition models used in the second decoding pass are usually triphones with a single Gaussian per state.

In the current version of the system, a projection of the acoustic feature space, based on Heteroscedastic Linear Discriminant Analysis (HLDA), is embedded in the feature extraction process as follows. A GMM with 1024 Gaussian components is first trained on an extended acoustic feature set consisting of static acoustic features plus their first, second and third order time derivatives. Acoustic observations in each, automatically determined, cluster of speech segments, are then normalized by applying a CMLLR transformation estimated w.r.t. the GMM. After normalization of training data, an HLDA transformation is estimated w.r.t. a set of state-tied, cross-word, gender-independent triphone HMMs with a single Gaussian per state, trained on the extended set of normalized features. The HLDA transformation is then applied to project the extended set of normalized features in a lower dimensional feature space, that is a 39-dimensional feature space.

Recognition models used in the first and second decoding passes are trained from scratch on normalized, HLDA projected, features. HMMs for the first decoding pass are trained through a conventional maximum likelihood procedure. Recognition models used in the second decoding pass are speaker adaptively trained exploiting, as seen above, as target-models triphone HMMs with a single Gaussian density per state.

2.3. Baseline LM

As previously mentioned the text data used for training the baseline LM are extracted from "google-news" web corpus. These data are grouped into 7 broad domains (economy, sports, science and technology, etc) and, after cleaning, re-

moving double lines and application of a text normalization procedure, the corpus results into about 5.7M of documents, for a total of about 1.6G of words. The average number of words per document is 272.

On this data we trained a 4-gram backoff LM using the modified shift beta smoothing method as supplied by the IRSTLM toolkit [14]. The LM results into about 1.6M unigrams, 73M bigrams, 120M 3-grams and 195M 4-grams.

As seen above the LM is used twice: the first time to compile a static Finite State Network (FSN) which includes LM probabilities and lexicon for the first two decoding passes. The LM employed for building this FSN is pruned in order to obtain a network of manageable size, resulting in a recognition vocabulary of 200K words, 37M bigrams, 34M 3-grams, 38M 4-grams. The non-pruned LM is instead combined (through equation 1) with the auxiliary LMs and used in the third decoding step to rescore word graphs.

2.4. Word graphs generation and rescoring

Word graphs (WGs) are generated in the second decoding step. To do this, all of the word hypotheses that survive inside the trellis during the Viterbi beam search are saved in a word lattice containing the following information: initial word state in the trellis, final word state in the trellis, related time instants and word log-likelihood. From this data structure and given the LM used in the recognition steps, WGs are built with separate acoustic likelihood and LM probabilities associated to word transitions. To increase the recombination of paths inside the trellis and consequently the densities of the WGs, the so called word pair approximation [15] is applied. In this way the resulting graph error rate was estimated to be around $\frac{1}{3}$ of the corresponding WER.

As shown if figure 1, for each given i^{th} talk an auxiliary LM (LM^i_{aux}) is trained using data selected automatically from a huge corpus (i.e. "google-news") with one of the methods described in section 3. The i^{th} query document used to score the corpus consists of the 1-best output of the second ASR decoding step, as depicted in Figure 1. Then, the original (baseline) LM probability on each arc of each WG is substituted with the interpolated probability given by equation 1. The interpolation weights, λ^i_{base} and λ^i_{aux}, associated to the two LMs (LM_{base} and LM^i_{aux}) are estimated so as to minimize the overall LM perplexity on the 1-best output (the same used to build the i^{th} query document), of the second ASR decoding step. For clarity reasons this latter procedure is not explicitly shown in Figure 1.

Finally, the rescored 1-best word sequences are used for evaluating the performance.

3. Auxiliary Data Selection

In this section we describe the processes for selecting documents (rows in "google-news" corpus, each one containing a news article) which are semantically similar to a given automatically transcribed document. In the following, N is the number of total rows of the corpus (5.7M for this work) and D is the total number of unique words in the corpus.

The result of this process is to obtain a sorted version of the whole "google-news" corpus according to similarity scores. The most similar documents will be used to build talk-dependent auxiliary LMs, trained on different amount of data.

3.1. TFxIDF based method

We are given a dictionary of terms $t_1, \ldots, t_D$ derived from the corpus to select (i.e. "google-news").

From the sequence of automatically recognized words $W^i = w^i_1, \ldots, w^i_{\mathtt{len}(W^i)}$ of the given i^{th} query document (i.e. the i^{th} automatically transcribed talk) the TFxIDF coefficients $c^i[t_d]$ are evaluated for each dictionary term t_d as follows [16]:

$$c^i[t_d] = (1 + \log(tf^i_d)) \times \log(\frac{D}{df_d}) \qquad 1 \leq d \leq D \quad (2)$$

where tf^i_d is the frequency of term t_d inside document W^i and df_d is the number of documents in the corpus to select that contain the term t_d.

The TFxIDF coefficients of the n^{th} row (document) in the "google-news" corpus $r^n[t_d], 1 \leq n \leq N$ are computed in the same way (where N is the total number of rows). Then, the two vectors $\mathbf{C}^i = c^i[t_1], \ldots, c^i[t_D]$ and $\mathbf{R}^n = r^n[t_1], \ldots, r^n[t_D]$ are used to estimate a similarity score for the n^{th} document via scalar product:

$$s(\mathbf{C}^i, \mathbf{R}^n) = \frac{\mathbf{C}^i \cdot \mathbf{R}^n}{\mid \mathbf{C}^i \mid\mid \mathbf{R}^n \mid} \quad (3)$$

The approach requires to evaluate N scalar products for each automatically transcribed talk. Each scalar product wants, according to equation above, to essentially compute Q^i_n sums plus Q^i_n multiplications, where Q^i_n is the number of common terms in W^i and in the n^{th} document, $W^n_{\mathtt{google-news}}$. Hence, the total number of arithmetic operations required for scoring the whole corpus is proportional to $O(2 \times N \times \mathtt{E}[Q^i_n])$, where $\mathtt{E}[]$ denotes expectation. Concerning memory occupation, the method basically requires to load into memory of the computer the IDF coefficients, i.e. the term $\log(\frac{D}{df_d})$ in equation 2, of all words in the dictionary, plus the $r^n[t_d]$ coefficients, for a total of $D + N \times \mathtt{E}[Q^i_n]$ float values. Then, TFxIDF coefficients of the query document are estimated through equation 2, while TFxIDF coefficients of each row of "google-news" are conveniently computed in a preliminary step and stored in a file. In our implementation, access to coefficients entering the scalar product of equation 3 is done using associative arrays. Note that we don't consider this contribution in the complexity evaluation of the approach.

Note also that sorting the whole corpus according to the resulting TFxIDF scores, to find out the most similar doc-

uments to the given query document talk, may be computationally expensive. Hence, we discard documents of the corpus whose TFxIDF scores are below a threshold and perform sorting only on the remaining set of documents. The latter threshold is determined through preliminary analyses of TFxIDF values, taking advantage from the fact that TFxIDF coefficients are normalized within the interval $[0-1]$.

3.2. New proposed approach

3.2.1. Preprocessing stage

First, we build a table containing all the different words found in the "google-news" corpus, each one with an associated counter of the related number of occurrences in the corpus itself. The words are sorted in descending order with respect to the counter and a list is built that includes only the most frequent D' words (in our case a choice of $D' = 200773$ allows to retain words having more than 34 occurrences). Then, from the resulting list the most frequent $D" = 100$ words are removed, allowing to create an index table, where each index is associated to a word in a dictionary $\mathcal{V}$ (lower indices correspond to words having higher counters). Finally, every word in the corpus is replaced with its corresponding index in $\mathcal{V}$. Words outside $\mathcal{V}$ are discarded. Indices of each row are then sorted to allow quick comparison (this point will be discussed later).

The rationale behind this approach is the following:

- very common words, i.e. those with low indices, only carry syntactic information, therefore they are useless if the purpose is to find semantically similar sentences;
- very uncommon words will be used rarely so they will just slow down the search process.

The choice for the reported values of D' and D'' has been done on the basis of preliminary experiments carried out on a development data set (see section 4) and resulted not to be critical. With the chosen values about half of the words of the corpus were discarded: currently there are 5.7 millions rows, corresponding in total to 1561.1 millions words, 864.5 millions survived indices. We keep alignment between the original corpus and its indexed version.

3.2.2. Searching stage

We apply to the given i^{th} talk the same procedure as before, obtaining a sequence of numerically sorted word indices. Hence, as for the TFxIDF method, both the i^{th} talk and the n^{th} "google-news" document are represented by two vectors (containing integer indices in this case): $\mathbf{C}'^i$ and $\mathbf{R}'^n$, respectively. The similarity score is in this case:

$$s'(\mathbf{C}'^i, \mathbf{R}'^n) = \frac{e(\mathbf{C}'^i, \mathbf{R}'^n)}{\mathtt{dim}(\mathbf{C}'^i) + \mathtt{dim}(\mathbf{R}'^n)} \qquad (4)$$

where $e(\mathbf{C}'^i, \mathbf{R}'^n)$ is the number of common indices between the two vectors $\mathbf{C}'^i$ and $\mathbf{R}'^n$.

Note that, differently from TFxIDF approach, where both vectors $\mathbf{C}^i$ and $\mathbf{R}^n$ can be assumed to have dimension equal to D (the size of the dictionary), in this case the normalization term for the similarity measure is given by the denominator of equation 4. The two vectors $\mathbf{C}'^i$ and $\mathbf{R}'^n$ have dimensions exactly equal to the number of the corresponding indexed words survived after pruning of dictionary, as explained above.

Note also that, while TFxIDF method allows to compare two documents by weighting same words both with their frequencies and with their relevance in the documents to select, the proposed approach is essentially a method to count the number of same words in the documents (word counters are not used in the similarity metric). However, since components of index vectors are numerically ordered, the computation of the similarity score $s'(\mathbf{C}'^i, \mathbf{R}'^n)$ results very efficient. This is essential given the large number of documents in the corpus to score.

Each of the N score computation, according to equation 4, essentially needs Q'^i_n comparisons (in this case no sums or multiplications are executed) to be executed, with $Q'^i_n \leq Q^i_n$, due to dictionary pruning. Since, we can assume $\mathtt{E}[Q'^i_n] \simeq \frac{1}{2}\mathtt{E}[Q^i_n]$ (due to halving of indices), the total number of comparisons required for scoring the whole corpus is proportional to $O(\frac{N}{2} \times \mathtt{E}[Q^i_n])$, i.e. $\frac{1}{4}$ with respect to TFxIDF based method. In addition, differently from the latter one, the proposed approach doesn't require to load into memory of the computer any parameter related to the whole dictionary, instead only the sequence of indices (i.e. one sequence of integer values for each row of "google-news") entering equation 4 is needed. In our implementation the latter indices are conveniently stored and read from a file. Therefore, the memory requirements of the proposed approach are negligible. Furthermore, since the resulting document scores are not normalized, the estimate of the threshold to be used for selecting the subset of the documents to sort from the whole corpus is based on a preliminary computation of a histogram of scores.

Finally, in order to measure the complexities of proposed method and TFxIDF based one, we led three different selection runs using ASR output of a predefined TED talk. For processing the whole "google-news" corpus the proposed method took on average about $16min$, with a memory occupation of about 10MB, while the TFxIDF based method took on average about $114min$, with a memory occupation of about 650MB. These runs were carried out on the same Intel/Xeon E5420 machine, free from other computation loads.

3.3. Perplexity based method

A 3-gram LM is trained with the automatic transcription of the given i^{th} TED talk. Then, the perplexity of each document in the "google-news" corpus is estimated using this latter LM and the resulting perplexity values are used to find out the most similar documents to the given talk. Also in this case an histogram of perplexity scores is es-

timated to determine the optimal selection threshold before sorting documents. Basically, each of the N perplexity values (one for each "google-news" document) requires to compute $\texttt{len}(W^n_{\texttt{google-news}})$ log-probabilities (through LM look-up table and LM backoff smoothing) and $\texttt{len}(W^n_{\texttt{google-news}})$ sums.

4. Experiments and results

As previously mentioned experiments have been carried out on the evaluation sets of IWSLT 2011 evaluation campaign. In total, these latter ones include 27 talks, which have been divided into a development set and a test set. Table 1 reports some statistics derived from evaluation sets.

Table 1: *Statistics related to the dev/test sets of IWSLT 2011 evaluation campaign: total number of running words, minimum, maximum and mean number of words per talk.*

	dev-set (19 talks)	**test-set (8 talks)**
#words	44505	12431
(min,max,mean)	(591,4509,2342)	(484,2855,1553)

Note the quite small number of words available for each talk to build the similarity models to be used in the automatic selection process, especially for the test set. Despite this fact, significant performance improvement has been achieved on this task.

We evaluated performance, both in terms of PP and WER.

The overall perplexity PP_{dev} on the dev set is computed summing the LM log-probabilities of each reference talk and dividing by the total number of words, according to the following equation:

$$PP_{dev} = 10^{\frac{\sum_{i=1}^{i=19} -\log_{10}(P^i_{LM}[W_i])}{NW}} \quad (5)$$

where $P^i_{LM}[W_i]$ is the probability of the reference word sequence in the i^{th} talk, computed using the i^{th} talk-dependent interpolated LM, and NW is the total number of words in the dev set. The overall perplexity on the test set is computed in a similar way.

Performance, as a function of the number of words used to train the auxiliary LMs, are reported in Figures 2 to 5, for both dev set and test set.

In the figures the point corresponding to 0 words on the abscissa indicates performance obtained using the baseline, talk independent, LM (i.e. no interpolation with auxiliary LMs has been made).

As can be observed all of automatic selection methods allow to improve both in terms of perplexity and WER. Looking at curves of perplexity (figures 2 and 4), we note that an optimal value for the number of words that should be used for training auxiliary LM is clearly reached with both TFxIDF and new proposed selection approach (the related curves

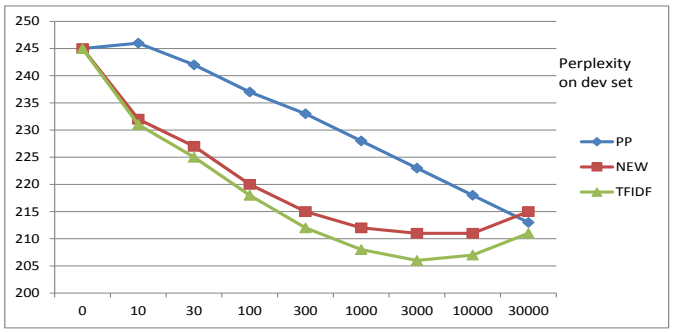

Figure 2: *Perplexity on dev set of PP-based selection method, NEW proposed method and TFxIDF based method as a function of the number of words, shown on a logarithmic scale, used to train the auxiliary LMs.*

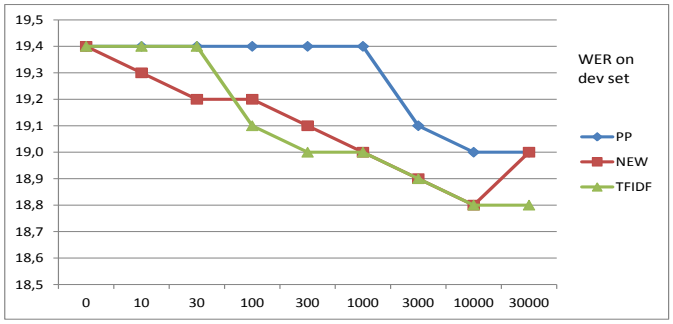

Figure 3: *%WER on dev set for the various selection methods.*

exhibit clear minimal points). Instead, this trend is not exhibited by PP based curves, where the minimal perplexity value seems will be reached with a quite high number of auxiliary words (we deserve to extend the curves with future experiments). This is probably due to the fact that proposed and TFxIDF selection methods give more weight to content words than the PP based one, where also functional words can significantly contribute to form the scores of documents to select.

A different trend is instead observed looking at curves related to WERs (see figure 3 and 5), specifically, they do not exhibit clear minimal values. Actually, while perplexity values depend only on LM probabilities (i.e. on models derived only from text data, including the selected ones), WER values are obtained through Maximum a Posteriori (MAP) decoding, combining LM probability scores and AM likelihood scores, giving rise to more irregularities in the related curves, as well as to local minima. In any case, it is important to note that the usage of focused LMs allow always to decrease WER. In particular, both new and TFxIDF approaches allow to achieve about 3% WER reduction on both

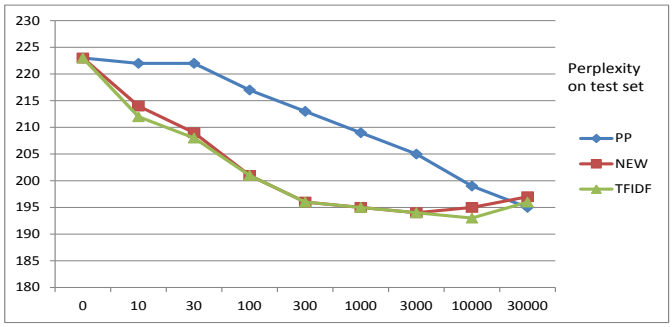

Figure 4: *Perplexity on test set for the various selection methods.*

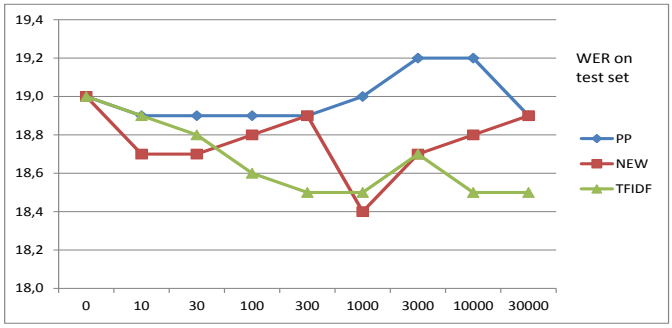

Figure 5: *%WER on test set for the various selection methods.*

dev and test sets, while a lower improvement (around 2% relative WER reduction) is obtained with the PP based selection method.

Finally, for comparison purposes we trained a domain specific LM using subtitles of TED talks that have been downloaded from the internet by the organizer of IWSLT 2011 evaluation campaign, before the cut-off date (December 31, 2010), and distributed to the participants. The latter domain specific corpus contains around 2M words and the resulting LM (LM_{ted}) contains about: 40K unigrams, 540K bigrams, 1.6M 3-grams and 1M 4-grams. Then, we have linearly interpolated LM_{ted} with the baseline LM (LM_{base}) and, as for the automatic selection methods, we have rescored the WGs generated in the second ASR decoding pass with the "adapted" LM (i.e. $LM_{base} \oplus LM_{ted}$, where symbol $\oplus$ denotes interpolation according to equation 1). Note that also in this case the linear interpolation weights have been estimated using the automatic transcriptions of the second ASR decoding pass. Table 2 reports the performance, both in terms of WER and PP, for the "focused" LMs (where LM_{pp}, $LM_{tf \cdot idf}$ and LM_{new} have been trained on automatically selected text corpora of 3M words) and for the domain adapted LM.

Table 2: *Results obtained using "focused" LMs and domain adapted LM.*

	dev-set		test-set	
	PP	**%WER**	**PP**	**%WER**
$LM_{base} \oplus LM_{pp}$	223	19.0	205	18.9
$LM_{base} \oplus LM_{new}$	210	18.8	194	18.4
$LM_{base} \oplus LM_{tf \cdot idf}$	206	18.8	194	18.5
$LM_{base} \oplus LM_{ted}$	158	18.7	142	18.4

As can be seen from the Table, although PP values for the domain adapted LM ($LM_{base} \oplus LM_{ted}$) are significantly lower with respect to the other LMs, the corresponding WER values are similar to those obtained with focused LMs. The proposed selection approach (row $LM_{base} \oplus LM_{new}$), gives 0.1% difference on the dev set and 0% on test set, respectively.

4.1. Experiments with IWSLT2012 data

To further check the effectiveness of LM focusing approaches described so far, we carried out additional experiments using the sets of English text corpora distributed for IWSLT 2012 Evaluation Campaign. These latter consist of: news commentaries and news crawls, proceedings of European Parliament sessions and the newswire text corpus Gigaword (fifth edition), as distributed by the LDC consortium (see LDC catalog http://www.ldc.upenn.edu/Catalog/ for more details about this corpus). In addition an in-domain text corpus containing transcriptions of TED talks has been provided.

With these data we built 3 LMs:

- LM_{W12}, trained on news commentaries/crawls and European Parliament proceedings (about 830M words);
- LM_{G5}, trained on Gigaword, fifth edition (about 4G words);
- LM_{T12}, trained on in-domain TED data (about 2.7M words).

Similarly to what reported in Table 2 we measured performance (both PP and WER) using talk-specific linearly interpolated LMs. In particular, we compared performance using different combinations of LMs, as shown on Table 3.

Also in this case talk-specific auxiliary LMs were trained on data (5M words) automatically selected using the ASR output of the second decoding step. The latter selection was carried out over both $W12$ and $G5$ text corpora (i.e. without using in-domain TED data). We only compared TFxIDF based method and the new one, proposed in this paper.

Table 3 gives the results on both development and test sets. In this case we haven't evaluated performance as a function of the number of words retained for auxiliary data selection (see figures 2 to 5). This latter number of words, according to previous experiments using IWSLT 2011 text data, has

Table 3: *Results obtained using baseline, "focused" and domain adapted LMs trained on text data delivered for IWSLT 2012 Evaluation Campaign.*

	dev-set		test-set	
	PP	%WER	PP	%WER
$LM_{W12} \oplus LM_{G5}$	179	18.8	159	18.1
$LM_{W12} \oplus LM_{G5} \oplus LM_{tf \cdot idf}$	155	18.4	140	17.6
$LM_{W12} \oplus LM_{G5} \oplus LM_{new}$	164	18.5	146	17.5
$LM_{W12} \oplus LM_{G5} \oplus LM_{ted}$	139	18.2	126	17.5

been fixed to 5 millions. Note also that with the new set of training text data the improvement given by the proposed focusing procedure is maintained (about 2% relative WER reduction on the dev set and about 3% WER relative reduction on the test set), performing very closely to domain adapted LMs.

5. Conclusions and Future Work

We have described a method for focusing LMs towards the output of an ASR system. The approach is based on the useful and efficient selection, according to a novel similarity score, of documents belonging to large sets of text corpora on which the LM used for automatic transcription was trained. Improvements on WER have been reached without making use of in-domain specific text data. In addition, comparisons with TFxIDF and PP based selection methods have been done, showing the effectiveness of the proposed approach, which resulted computationally less expensive than TFxIDF.

However, at present we are not able to decide if this result is quite general, or if it depends on the particular set of data used, or on the specific TED domain. Future works will try to extend the approach to domains different from TED.

6. References

[1] J. Gao, J. Goodman, M. Li, and K. Lee, "Toward a Unified Approach to Statistical Language Modeling for Chinese," *ACM Transactions on Asian Language Information Processing*, vol. 1, no. 1, pp. 3–33, 2002.

[2] D. Klakow, "Selecting Articles from the Language Model Training Corpus," in *Proc. of ICASSP*, Istanbul, Turkey, June 2000, pp. 1695–1698.

[3] R. C. Moore and W. Lewis, "Intelligent Selection of Language Model Training Data," in *ACL Conference*, Uppsala, Sweden, July 2010, pp. 220–224.

[4] S. Maskey and A. Sethy, "Resampling Auxiliary Data for Language Model Adaptation in Machine Translation for Speech," in *Proc. of ICASSP*, Taipei,Taiwan, April 2009, pp. 4817–4820.

[5] G. Lecorve, J. Dines, T. Hain, and P. Motlicek, "Supervised and unsupervised Web-based language model domain adaptation," in *Proc. of INTERSPEECH*, Portland, USA, September 2012.

[6] K. Thadani, F. Biadsy, and D. M. Bikel, "On-the-fly Topic Adaptation for You Tube Video Transcription," in *Proc. of INTERSPEECH*, Portland, USA, September 2012.

[7] X. Liu, W. Byrne, M. Gales, A. de Gispert, M. Tomalin, P. Woodland, and K. Yu, "Discriminative Language Model Adaptation for Mandarin Broadcast Speech Transcription," in *Proc. of ASRU*, Kyoto, Japan, December 2007, pp. 153–158.

[8] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Cernocky, "Empirical Evaluation and Combination of Advanced Language Modeling Techniques," in *Proc. of INTERSPEECH*, Florence, Italy, August 2011, pp. 605–608.

[9] N. Ruiz, A. Bisazza, F. Brugnara, D. Falavigna, D. Giuliani, S. Jaber, R. Gretter, and M. Federico, "FBK@IWSLT 2011," in *Proc. of IWSLT workshop*, San Francisco, USA, December 2011.

[10] D. Giuliani and F. Brugnara, "Experiments on Cross-System Acoustic Model Adapatation," in *ASRU Workshop 2007*, Kyoto, Japan, Dec. 2007, pp. 117–122.

[11] J. Loof, D. Falavigna, R. Schluter, D. Giuliani, R. Gretter, and H. Ney, "Evaluation of Automatic Transcription Systems for the Judicial Domain," in *Proc. of IEEE SLT workshop*, San Francisco, USA, December 2010, pp. 194–199.

[12] M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech and Language*, vol. 12, pp. 75–98, 1998.

[13] G. Stemmer, F. Brugnara, and D. Giuliani, "Using Simple Target Models for Adaptive Training," in *Proc. of ICASSP*, vol. 1, Philadelphia, PA, March 2005, pp. 997–1000.

[14] M. Federico, N. Bertoldi, and M. Cettolo, "IRSTLM: an Open Source Toolkit for Handling Large Scale Language Models," in *Proc. of INTERSPEECH*, Brisbane, Australia, September 2008, pp. 1618–1621.

[15] X. Aubert and H. Ney, "A word graph algorithm for large vocabulary continuous speech recognition," in *Proc. of ICSLP*, 1994, pp. 1355–1358.

[16] J. Ramos, "Using TF-IDF to Determine Word Relevance in Document Queries," in *First International Conference on Machine Learning*, New Brunswick:NJ, USA, 2003.