Combination of Random Indexing based Language Model and N-gram Language Model for Speech Recognition

Dominique Fohr 1,3, Odile Mella 2,3

1 CNRS, LORIA UMR 7503, Vandoeuvre-lès-Nancy, F-54506, France
2 Université de Lorraine, LORIA UMR 7503, Vandoeuvre-lès-Nancy, F-54506, France
3 Inria, Villers-lès-Nancy, F-54600, France
Dominique.Fohr@loria.fr, Odile.Mella@loria.fr

Abstract

This paper presents the results and conclusion of a study on the introduction of semantic information through the Random Indexing paradigm in statistical language models used in speech recognition. Random Indexing is an alternative to Latent Semantic Analysis (LSA) that addresses the scalability problem of LSA. After a brief presentation of Random Indexing (RI), this paper describes, different methods to estimate the RI matrix, then how to derive probabilities from the RI matrix and finally how to combine them with n-gram language model probabilities. Then, it analyzes the performance of these different RI methods and their combinations with a 4-gram language model by computing the perplexity of a test corpus of 290,000 words from the French evaluation campaign ETAPE. Among the results, the main conclusions are (1) regardless of the method, function words should not be taken into account in the estimation of RI matrix; (2) The two methods RI basic and TTRI. w achieved the best perplexity, i.e. a relative gain of 3% compared to the perplexity of the 4-gram language model alone (136.2 vs. 140.4).

Index Terms: speech recognition, random indexing, language modeling

1. Introduction

This paper addresses the issue of including more semantic information in automatic speech recognition by using the Random Indexing (RI) paradigm in language modeling. Language modeling plays an important role in automatic speech recognition (ASR) because it constrains the decoder to search the most likely sequences of words according to a given language and a given task.

For decades, statistical N-gram language models have successfully fulfilled this role. But they model only local dependencies and they are currently not able to capture large dependencies and they are currently not able to capture large medical data bases [11] and [12]. But to the best our knowledge, Random Indexing has not been applied to speech recognition. Therefore, we wanted to study whether it was possible to use Random Indexing in language modeling for speech recognition.

This paper is organized as follows: in Section 2, we introduce the Random Indexing (RI) paradigm and the training methods we used. In Section 3, we present how the information coded in the RI matrix can be converted into probabilities and combined with n-gram probabilities. In Section 4, we describe our evaluation methodology and in Section 5 we present and discuss experimental results and finally we draw some conclusions.

2. Random Indexing

2.1. Basic Random Indexing

Random indexing (RI) is a scalable alternative to LSA for analyzing relationships between a set of documents and the terms they contain. Given a set of D documents and a set of T terms, the underlying principle to the RI is to reduce the dimensionality of the co-occurrence matrix \( F_{DxT} \) by multiplying it with (or projecting it through) a random matrix \( R_{DxK} \), \( R_{TxD} \) with \( K \ll D \). This method rests on the Johnson-Lindenstrauss lemma [13] that states that if we project points in a vector space into a randomly selected subspace of sufficiently high dimensionality, the distances between the points are approximately preserved. The random index vectors of the matrix \( R \) must be nearly orthogonal. An index vector with small number of randomly distributed +1s and -1s, and the rest of the elements of the vectors set to 0, respects this constraint.

In its basic form, RI involves two steps. In the first step, the components of the index vector of every document are initialized to 0 except a small number of them randomly set to +1 or -1. All the term index vectors are set to 0. Then, in the training step, the index vector of each term is obtained by adding the document index vector every time the term occurs in this document. Prior to addition, the document index vector can be weighted by a factor which depends on the term and the document.

In the following sections, the set of the document index vectors is named \( document \) matrix and the set of the term index vectors, \( term \) matrix.
2.2. Different methods of Random Indexing

As Cohen and al. in [11], we have investigated several methods for training Random Indexing matrices:

- **RI basic**: the RI method described Section 2.1:

  \[
  \text{Document index vector} \quad \text{Term index vector} \\
  \text{Doc 1: 0, 0, 0, 1, 0, 1, 1, ...} \\
  \text{Doc 2: 1, 0, 0, 1, 0, 1, 0, ...} \\
  \text{Doc 3: 0, 0, 0, 1, 0, 0, 1, ...} \\
  \text{Term 1: 1, 0, 1, 1, 0, 0, 0, ...} \\
  \text{Term 2: 0, 1, 0, 1, 1, 0, 1, ...} \\
  \text{Term 3: 0, 0, 0, 1, 0, 0, 0, ...} \\
  \text{Term 4: 0, 1, 0, 0, 1, 1, 0, ...} \\
  \text{Term 5: 0, 0, 0, 1, 0, 0, 0, ...} \\
  \text{Term 6: 0, 0, 0, 0, 0, 1, 1, ...} \\
  \text{Term 7: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \]

- **DRRI** (Document-based Reflective RI): after both steps of RI, a new document matrix is computed from the term matrix. The document index vector is the weighted sum of the index vectors of all the terms of the document. Then, a new term matrix is computed from this document matrix using RI:

  \[
  \text{Document index vector} \quad \text{Term index vector} \\
  \text{Doc 1: 1, 0, 0, 0, 1, 1, ...} \\
  \text{Doc 2: 0, 1, 0, 0, 1, 0, ...} \\
  \text{Doc 3: 0, 0, 1, 0, 1, 1, ...} \\
  \text{Term 1: 0, 0, 1, 0, 0, 0, ...} \\
  \text{Term 2: 1, 1, 0, 0, 0, 0, ...} \\
  \text{Term 3: 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 4: 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 5: 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 6: 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 7: 0, 0, 0, 0, 0, 0, ...} \\
  \]

- **TRRI** (Term-based Reflective RI): in the first step, the components of the index vector of every term are initialized to 0 except a small number of them randomly set to +1 or -1. All the document index vectors are set to 0. Then, as above the document matrix is computed from the term matrix and finally, a new term matrix is computed from the document matrix:

  \[
  \text{Term index vector} \quad \text{Document index vector} \\
  \text{Term 1: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 2: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 3: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 4: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 5: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 6: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 7: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Document index vector} \quad \text{Term index vector} \\
  \text{Doc 1: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Doc 2: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Doc 3: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 1: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 2: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 3: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 4: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 5: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 6: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \text{Term 7: 0, 0, 0, 0, 0, 0, 0, ...} \\
  \]

2.3. Weighting functions

We have tested several weighting functions. The simplest function is a uniform function for each term (identity) but it could give excessive weight to frequent terms. Weighting by the inverse of the term frequency computed on the entire collection of documents (freq) avoid this drawback. We have also tested the inverse of the logarithm of the frequency (logfreq). A more sophisticated function (logentropy) takes into account the frequency of the term in a document and the frequency of the term within all the documents (D):

\[
P_{\text{at}} = \frac{\text{frequency of term } t \text{ in document } d}{\text{frequency of term } t \text{ in all documents}} \\
global\_weight(t) = 1 + \sum_d \frac{\log(p_{\text{at}})}{\log(D)} \\
\]

\[
Log\_ent\_\text{ropy}\_\text{rate}\ = -\sum_{t,d} p_{\text{at}} \log(p_{\text{at}}) \\
\]

3. RI based language model

To exploit the semantic information contained in the RI matrix in the framework of speech recognition, it is necessary to estimate a semantic probability between the word to be recognized and its context. Then, this RI probability will be combined with the N-gram probability of the word.

3.1. Deriving RI Probabilities

Given a RI term matrix, we assumed that two terms are semantically close if the cosine of the angle of the two corresponding term index vectors is close to 1. In the same manner, a word is semantically close to a context (a bag of words) if the cosine between the term index vector and the context index vector is close to 1. The context index vector is the sum of the term index vectors of all the words of the context. In speech recognition, the context is either all other words of the recognized sentence (post-processing) or the words of the sentence already processed (during the recognition step).

Before integrating the cosine into the speech recognition process or in the computation of the perplexity, it is necessary to convert it into a probability. For that we used the method proposed by Coccaro and Jurafsky in [5] for LSA.

First, given a context C, we computed the smallest cosine between the context vector C and any term t of the set T:

\[
\text{MinCos}(C) = \min_{t \in T} \cos(C, t) \\
\]

Finally, to increase the range of the estimated probabilities, the probability was raised to a power γ:

\[
P_{\text{RI}}(t|C) = \frac{\text{MinCos}(C) - \text{MinCos}(C)}{\sum_{t \in T} \text{MinCos}(C) - \text{MinCos}(C)} \\
\]

3.2. Combining RI and N-grams probabilities

Statistical N-gram language models and RI language model provide additional information. The RI language model predicts a global semantic relationship between a word and the other words of the sentence but it provides no information about the gender and the number of the word nor about its exact location in the sentence. Furthermore, as function words contain no semantic information, they cannot be modeled by the RI language model. Otherwise, N-gram language models can model function words, and gender and number agreement but only locally. Thus it is necessary to combine the probabilities estimated by both language models. We chose the geometric combination because it has achieved better results than linear combination for LSA [5].

\[
P(w, C) = \frac{P_{\text{RI}}(w|C)^{\lambda(w)}}{\sum_{t=1}^{D} P_{\text{RI}}(w|C)^{\lambda(w)}} \\
\]

In the above equation, w is the word that we want to estimate the probability, C is the context: all the words of the sentence except itself, H is the n-gram history: the (n-1) words preceding w, \( P_{\text{NG}} \) is the n-gram probability, \( M \) the number of words in the n-gram model and \( \lambda(w) \) is the geometric weight.

We chose a coefficient \( \lambda(w) \) which depends on the word w for two reasons. Firstly, if the word w is not a term in the RI
matrix, we set $\lambda(w)$ to 0. Secondly, we assumed that the RI probability of words belonging to few documents is more relevant. As these words should have a high $global\_weight$, we chose a $\lambda(w)$ based on the $global\_weight$ (cf. Eq. 2).

### 4. Experimental conditions

#### 4.1. Training of RI model

The different RI matrices were trained on the articles of the newspaper “Le Monde” between 1987 and 2005, i.e. a total of 390 million of words. Each article was considered as a document for the training of RI matrices, i.e. 894,000 documents (value of $D$). Because only the semantics of a word is important, every word of a document is replaced by its lemma as given by the BD Lex dictionary [16].

#### 4.2. Training of N-gram model

We used the SRI Toolkit [15] to estimate a statistical 4-gram on a corpus of French broadcast news transcriptions. We chose this corpus of 100 million of words because it is similar to the test corpus (recording conditions, topics, style of speech).

#### 4.3. The lexicon

The lexicon defines the terms used for estimating the RI matrices and the vocabulary of the N-gram language model. We designed a baseline lexicon of 96,000 spelling entries which are the most frequent words in training corpora. This baseline lexicon was lemmatized for RI.

#### 4.4. Evaluation methodology

We tested our different RI models and combinations by computing the perplexity on a test corpus of 290,000 words. It consists of orthographic transcriptions of a part of the ETAPÉ corpus. The ETAPÉ corpus, designed for the French evaluation campaign ETAPÉ [14], is composed of radio and TV broadcast news and debates, amusements and shows.

Unlike the RI train corpus that was split into documents, the test corpus is split into sentences. Thus, the context taken into account for estimating the RI probability of a word is made of all the other words of the sentence. This context, shorter than that used in the training phase, penalizes the estimation of the semantic probability but it is more realistic within the framework of a speech recognition system processing utterance by utterance. We divided the test corpus into 10 chunks of about 2000 sentences each. The perplexity of the 4-gram language model computed on every chunk of the test corpus varies from 102.9 to 241.5 (cf. Figure 2). The different chunks do not have the same complexity because the test corpus varies from 102.9 to 241.5 (cf. Figure 2). The different chunks do not have the same complexity because some of them contain more spontaneous speech (debates, shows). In the following tests, we essentially used the chunk with the highest perplexity: chunk #1.

For the sake of notation, whenever we indicate the name of a RI method that implies that it was combined with the 4-gram model according to the Equation 7.

### 5. Experimental results

#### 5.1. Impact of function words in Random Indexing

We consider that Random Indexing cannot be applied to function words as pronouns, adverbs, conjunctions, determiners, interjections, prepositions and some words such as day of the week or modal verbs. Indeed, function words carry very limited semantic information. But, overall, a function word occurring in two training documents without any semantic link will make the other words of both documents closer by almost RI methods.

In a first experiment, we wanted to check this assumption. Table 1 shows the perplexity computed on the test chunk #1 on the one hand, by the 4-gram language model alone and, on the other hand, by the combination of the 4-gram and RI models.

#### Table 1: Perplexity for the test chunk #1 according to RI matrices with and without function words

<table>
<thead>
<tr>
<th>Method</th>
<th>RI matrices without function words</th>
<th>RI matrices with function words</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-gram only</td>
<td>241.5</td>
<td>241.5</td>
</tr>
<tr>
<td>RI basic</td>
<td>228.1</td>
<td>255.7</td>
</tr>
<tr>
<td>TTRI w7</td>
<td>228.3</td>
<td>245.8</td>
</tr>
<tr>
<td>DRRRI</td>
<td>237.9</td>
<td>245.8</td>
</tr>
<tr>
<td>TRRI</td>
<td>234.0</td>
<td>245.4</td>
</tr>
</tbody>
</table>

Regardless of the RI method and its parameters, when function words are taken into account for estimating RI matrices and probabilities, the combination of both language models is worse than the 4-gram language model alone. Therefore, in the following experiments, the function words have been excluded for training RI matrices.

#### 5.2. Evaluation of the RI methods

This section presents the results of the test of the different parameters of the RI methods: the algorithm (RI, DRRRI, TTRRI), the weighting function ($identity$, $freq$, $logfreq$, $logentropy$), the size of the index vector (500, 1500, 2500, 3500, 5000, 10000, 15000, 20000, 30000) and the number of initial non-zero values of the index vector (20, 40, 80, 200, 400, 600). These parameters were assessed by computing perplexity on the chunk #1.

We observed that the size of the index vector and the number of initial non-zero values have almost no influence on the perplexity. Accordingly, we chose for the following experiments 2500 as index vector size and 20 as number of initial non-zero values.

As regards the best weighting function, it changes with the RI method:

- For RI basic, by definition, the weighting achieved by $identity$, $freq$ and $logfreq$ is identical because these functions do not depend on the document. Moreover, $logentropy$ and $identity$ give similar perplexity results.
- For TTRI w7, $logentropy$ is slightly better than the other three.
- For DRRRI, $logentropy$ provides best results.
- For TRRI, $identity$ provides best results.

The $freq$ function gives the worst results for DRRRI and TRRI methods. This can be explained by the fact that some very frequent words are nevertheless semantically relevant as for instance *France, président, américain* and *gouvernement* in our training corpus. So, their relative weight in the calculation of the index vector of a document should not be reduced.

#### 5.3. Impact of $\gamma$ in RI probability

We have introduced $\gamma$ in the calculation of the RI probability (cf. Eq. 6) in order to increase the range of the RI probabilities and to balance the RI and 4-gram probabilities. Figure 1 shows the perplexity computed on the chunk #1 according to $\gamma$ for every RI method. For each RI method, we selected the weighting function providing the best perplexity. For comparison, the perplexity of the 4-gram language model is
also plotted. We can observe that for all the RI methods, the optimal value of $\gamma$ is around 40. Moreover, two methods are much better than the others: RI_basic and TTRI_w7.

5.5. Optimization of the geometric combination

As explained in Section 3.2, the geometric weight $\lambda(w)$ is a function of the global_weight:

$$\beta$$

is a word-independent weighting factor whose role is to balance the contribution of 4-gram and RI language models. Indeed, both models are complementary. The 4-gram takes into account the order of the words and the local agreement in gender and number. The RI model captures larger span dependencies between words semantically related.

Figure 4 shows the impact of the weighting factor $\beta$ for the different methods RI. Best perplexity is achieved with the values of $\beta$ between 1/3 and 1/2. We can notice that in [5] for LSA, Coccaro arbitrarily chose 1/2 for $\beta$.

Given that the average value of the global_weight for all words of the lexicon is 0.6, the best weighting for the RI model, $\lambda(w)$, is between 0.2 and 0.3. This is consistent with the fact that N-gram model is still a good predictor.
7. References


2236