SOM network-based language model.

Leszek Gajecki

University of Information Technology and Management
Ul. Sucharskiego 2, 35-205 Rzeszów

Abstract

The Language Model module is a part of the Large Vocabulary Continuous Speech Recognition System (LVCSR). N-gram model and its derivates are the typically applied solutions. They are simple and computationally fast. Some improvements in these models help in solving the problem of data sparseness. This solution is well suited for languages with strict order of words (for example English), other languages, however, require more training examples. The proposed solution is a model based on Kohonen's Self Organised Maps, which smooth probabilities, but also can be used for languages with less strict order of words (e.g. Polish). The presented results show substantial improvement in comparison to trigram and bigram models in dictation task.

Keywords: language modelling, speech recognition, self organised maps
2010 MSC: 00-01, 99-00

1. Introduction

The presented work [1] describes a novelty in the field of language modeling for a Large Vocabulary Continuous Speech Recognition (LCVSR). The main purpose of the Language Model module application in such systems is to improve the quality of speech recognition measured by the Word Error Rate (WER). It is done by evaluation which word sequences constructed by LVCSR system can

Email address: lgajecki@wsiz.rzeszow.pl (Leszek Gajecki)

[1] This work was supported by the grant from the National Science Centre DEC-2011/03/B/ST7/02518 and by Podkarpackie Voivodship Scholarship Fund.

be sentences of spoken language. They should be linguistically correct, but this module need to be flexible for new phenomenons for living language, since daily-used language is changing. In spoken language also simplifications are present.

In this work the dictation task in the Polish language is considered (moreover, it is constrained to the spoken language and to limited range of vocabulary). The important property of the Polish language is its flexible word order, which, on the other hand, it is not a completely free order. One of the commonly applied solutions is the n-gram model [1, 2] and especially its improvements. Pure n-gram model solves the problem of data sparseness: many n-tuple of words occur either few times in a training corpus, or they are not present, which causes that respective probabilities are not correctly modeled. The second property is the importance of the word order. It is true for this base model, and for the majority of its derivate models. The first problem can be partially solved by the class-based ngram model [2], or the backoff n-gram model [3]. In a similar way, the Weighted Finite State Transducer [1] is sensitive to the word order. Consideration of some methods, that could improve trigram model for Polish are presented in [4].Problem of flexible (or free) word order can be solved by the Finite-State Grammar, which brings good results [5, 6]. However, the need for manual preparation of such grammar limits their application to the case of simple grammars.

Another solution could be the Head-Driven Phase Structure Grammar (HPSG) [7, 8]. This is constraint -based formalism. It consists two parts. The first one is a small amount of constrains which are here general rules. The second part is large amount of lexical entries, which define word-specific rules. The HPSG is very strict formalism. It has two consequences. First one is lack of needed language simplifications modeling, which are present in spoken language. Secondly, in order to obtain enough complex grammar a huge amount of manual work is needed to create such a lexicon. To solve presented problems author proposed simple Shallow Grammar based on HPSG rules in [9]. In that work partial parsing of sentence was introduced. One should mention that typically speech recognition is not considered as a field of Shallow Grammar application [10].

2. Contribution of presented work

In this paper author proposes language modeling on the base of Self Organized Memory (SOM) to solve two problems: the data sparseness and to improper modeling of language with flexible word order. To the best author's knowledge there is no similar solutions (except author's ones) that applies SOM. The novelty is also the application of wider input context (in quantity of words), than the input of the network. It also introduces new way of final output calculation based on several results given by this network. The next contribution of this work is the word probability determination as a single result, based on this network neurons weights and coverage them by the data. This work is continuation of author's PhD thesis [2] [11]. Here the feature selection will be considered in connection to improved training procedure.

The paper is organized as follows: in order to explain the place of the language model application, the architecture of typical LVCSR system is presented. Next, the SOM network is introduced and some solutions that let to create Language Model module are explained. Some examples will illustrate that such module can learn by examples the connections between words. The training process with feature selection is described. All in all, the results of experiments are presented with their discussion.

3. Architecture of typical LVCSR system

This section will explain the role of Language Model module in a LVCSR system. The most common is the architecture based on Hidden Markov Models (HMM's) [12]. The general schema of such system (Fig 1) is applied in HTK

[2] PhD thesis [11] got distinction in the 2014 Polish Artificial Intelligence Society award for the best Artificial Intelligence PhD thesis. PAIS is the ECAI member.

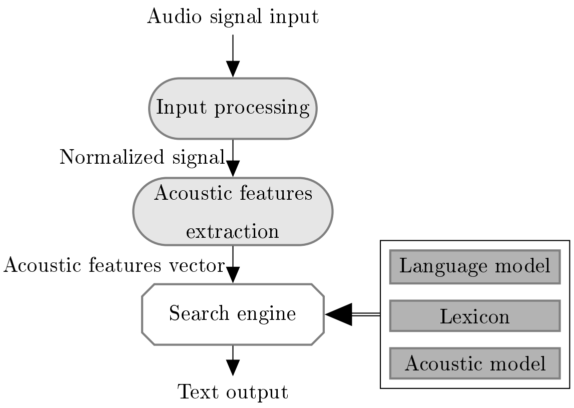


Figure 1: Architecture of a typical LVCSR system.

system [13], or the ESAT [14] etc. It is also present in systems applied for speech recognition in Polish Language [5, 15].

Input processing module performs signal normalization, while the output of *feature extraction* module is acoustic features vector. The *search engine* prepares speech hypothesis, using above vectors and knowledge from:

- *acoustic module* which models basic speech units like phonemes, triphones
- *lexicon*, which represents the construction of words using above basic speech units. Words as an symbols are choosen on the basis of their occurience in corpora. However mapping them into phonemes is mostly predefined.
- *language model*, which evaluates the possibility of occurrence of a given word sequence in considered language.

The Language Model can allow for one of them:

- some connections between words only
- or allows for all possible connections and returns probability measure, telling how probable is this hypothesis

After the construction of the speech hypotheses, the search engine performs searching for the best scored one W_{opt}, which is supposed be the closest to spoken utterance.

The search space covers all possible word sequences $W_1^K = w_1, w_2, \ldots, w_K$, where the acoustic vector $X = [x_1, x_2, \ldots, x_n]$ is given To preform this, to each hypothesis the cost is assigned:

$$W_{opt} = \arg\max_{W_1^K} P(W_1^K|\mathbf{X}) \tag{1}$$

The cost of each hypothesis:

$$\begin{aligned} f\left(\mathbf{X}, W_1^K\right) = &\log\ P\left(\mathbf{X}|W_1^K\right) + \\ &CA\ \log\ P\left(W_1^K\right) + CC \end{aligned} \tag{2}$$

where:

CA - is weight of probability $P(W_1^K)$ obtained from language model,

CC - its negative value is interpreted as penalty for starting a new word,

$\mathbf{X}$ - acoustic feature vector,

W_1^K - word sequence,

$P\left(\mathbf{X}|W_1^K\right)$ - probability given by acoustic model.

Resulting sequence of words is returned to textual output. One can say, that we check the similarity of acoustic units to speech signal, but possible constructions are limited by lexicon, while the language module refine this searching preferring word sequences possible in language (but letting for some exceptions).

There exists some other approaches to general LVCSR architecture. First one is expansion by adding the semantic module [12, 16, 17]. Noteworthy solution is words acquisition in project ACORNS [18], where the creators argue with traditional predefined lexicon. They let the system to learn words and word-like units (which can be parts of words for example). As we can notice (and briefly explained above in the definition of that module), the Lexicon is not trained in opposiotion to Acoustic Model and Language Model.

4. Related works

At the section 1 there were mentioned language models, that are improved n-gram models. Here will be briefly described the other approaches, that brings improvement for speech recognition, mostly using neural networks. N-gram model is interpolated with some of them.

Work [19] presents the application of a single layer forward network with the softmax function. The input refers to $k-1$-th word in word sequence (context length is 2). For a word with index i-th from the lexicon the i-th input is 1 and the other inputs are 0. The k-th word in considered word sequence (current word), is described by outputs. The word with index j-th from lexicon is refered by the j-th output. This value is interpreted as the conditional probability for the j-th word. Work [20] employs also coding function of a one word, which mapping is trained together with neural network to minimize perplexity. The input to a multilayer perceptron network is the concatination of two words coding. These words are coming from the context. First layer is hidden ($tanh$ function) and the second one is the layer with softmax function for normalisation purposes. The recurrent networks can be also used for this purpose, since they remember longer context (theoretically unlimited, but the influence of words in distant positions is relatively small). In the work [21] the structure of network is similar to above work, but words in a context are inputs that was send some steps before the current word. The example of recurrent networks was also applied to the Polish Language LVCSR [22] (called here Long-Short Term Memory). The speedup method for training and recognition by hierarchical decomposition is shown in [23].

Interesting approach of using the language structure is Structured Language Model [24], which is syntax-based language model. Words from not long context (3-5) are parsed (partially) according to language rules and resulting probabillity bases on some probabilities according to rules that build parsing tree. Improved model is Probabilistic Left Corner Parser [25]. The use of neural network for determination probabilities associated to such rules is proposed in [26].

Self-Organizing Maps have also been applied from the semantic modeling [27] for simple Language, where the animals were clustered according to their properties. Author proposes also complex neural networks application [28]. Further information about neural network application can be found in [29, 30].

5. Neural network

5.1. Introduction

The neural network is applied here to find association between grammatical classes of words, which can be done by unsupervised learning of Self-Organized Maps (SOM). The classes, to which given words belong, are defined in the lexicon. Each class is considered here as a Part of Speech (POS) and they are defined in IPI PAN corpus of Polish by thirteen properties: flexeme, number, case, gender, person, degree, aspect, negation, accent, post-prepositionality, accommodability, agglunativeness, vocalicity.

The author proposed already application of single SOM network for language modeling [31]. The mentioned hierarchical neural network model was described in [28]. The model presented here uses single SOM network, but in comparison to that works, new approaches for input and output processing were developed. To introduce such model the properties of SOM network as its central unit, will be reminded firstly.

5.2. SOM network

The SOM network [29] contains N neurons in the single layer. The inputs should be normalized, so that their norms are equal 1:

$$\|\mathbf{x}\| = 1 \tag{3}$$

where the norm is:

$$\|\mathbf{x}\| = \sqrt{\sum_{i=0}^{M} x_i^2} \tag{4}$$

The outputs are calculated in three possible ways (each one will be examined):

- Using a Hamming distance. The distance between input vector and weights, that defines the j-th neuron:

$$d = \sum_{i=1}^{N} |\mathbf{x}_i - \mathbf{w}_{ij}| \tag{5}$$

  One can construct the output function, which the maximal value will be equal 1, and it will be dimnishing (where β defines the its slope):

$$\mathbf{y}_j = \frac{\beta}{\beta + d} = \frac{\beta}{\beta + \sum_{i=1}^{N} |\mathbf{x}_i - \mathbf{w}_{ij}|} \tag{6}$$

- Using an Euclidean distance:

$$d = \sqrt{\sum_{i=1}^{M} (\mathbf{x}_i - \mathbf{w}_{ij})^2} \tag{7}$$

  In the similar way one can write the output function:

$$\mathbf{y}_j = \frac{\beta}{\beta + d^2} = \frac{\beta}{\beta + \sum_{i=1}^{M} (\mathbf{x}_i - \mathbf{w}_{ij})^2} \tag{8}$$

- scalar product (plus one):

$$\mathbf{y}_j = \sum_{i=1}^{N} \mathbf{x}_i \mathbf{w}_{ij} + 1 \tag{9}$$

Learning is performed using Winner Takes Most strategy. To inputs we put the training data. The neuron, which activation is the highest is called winner. Such neuron get the highest impact of this learning step while the neighboring neurons get smaller change of their weights. Neighbor function determines the influence on given neuron. In this work the neighborhood is linear, closed. Oja's rule is applied:

$$\Delta w_{ij} = y_j (x_i - w_{ij}) \eta \alpha(t) h(j, win) \tag{10}$$

where:

$\mathbf{w}$ – weights,

$\mathbf{x}$ – input,

$\mathbf{y}$ – output,

η – learning rate,

$\alpha(t) = \frac{\alpha_0 t}{C+t}$ – diminishing function depends on learning time (t-the number of learning steps). Such a slope for the function is necessary to make learning stable.

$h(j, win) = e^{-|j-win|}$ – a neighbor function,

win – number, index of the winning neuron (linear structure of a network),

j -– number, index of current neuron.

In this point we can put several questions, which help in formulation of model:

- What is the interpretation of the output of network?
- How to determine the language model output as a word probabilities?
- How the data can be mapped to the input?
- How to consider the relationships between non-adjacent words in sentence using this network?

5.3. Input coding

In this work we consider following coding. This is the vector of real numbers, each number represent one property, fixed coding. Numbers was chosen in arbitrary order, equal intervals, with maximal value equal 1. Similar coding is applied in [19]. It is worth to notice that in the work [28] author considered also other way of coding, however the choice of relevant features should be discussed now. In that work the six features were selected based on author's linguistic knowledge. In this work the feature analysis will be preformed. Since the feature vector consist real numbers, the Principal Component Analysis can be taken.

During experiments we decide how much features will be send to the network.

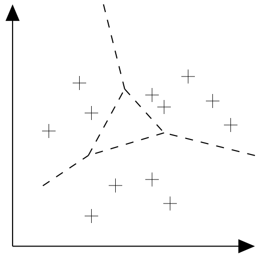


Figure 2: The partitioning of the data space into positive and negative clusters

6. Neural language model

6.1. Clustering

This network shows to which cluster belongs the data represented on the input. However there is the need to make that the result of this network work a classification: if the given utterance is valid or not. Once the network learning process is finished, one counts the coverage. This is a number of times each neuron in the network is active when using all elements from the learning set. Correct grammar relations are represented by neurons which wins relatively often (positive clusters), while incorrect rules refers to other neurons -negative clusters (Fig. 2).

The index *win* of winner neuron indicate if given words belong to learned language rule. It depends if such neuron's coverage is greater that threshold *covThreshold*. This gives us an opportnity to create function that calculate the language ,odel result. Let we refer to eq. 2. In order to use this network as a language model we don't need 2-valued function, but we need such one which can differ parts of sentences : bit better / bit worse.

In comparison of two rules when we have more frequent one , words might be accepted even if recognized with not so high acoustic probability (which occurs i.e. in case of bigger differencies to trainig data, may be not perfect pronounciation). At other hand, words recognized as very sure, quite clean, might be connected with not so frequent rule.

We need to notice also a computational problem: minimal returned value by Language Model shouldn't be zero, since its logarithm would be equal $-\infty$ and cause the throwing such hypothesis without consideration of acoustic models, or rules satisfied by parts of this sequence other than current words. In this case we will have at least p_{zero} minimal value, which can be done by max function:

$$p = max\,(p_{zero}, \dots) \tag{11}$$

To utilize an information on how strong the detected association is, the author proposes three kind of functions:

1. *Quantity of the cluster.* It can be also compared with probability of word class 1 in condition of word class 2.

   The cluster, to which belongs the input $\mathbf{x}$, is identified by neuron win. The proportion of such cluster coverage to all clusters' coverage: $\frac{cov(win)}{\sum_j cov(j)}$, where

   win – index of winner neuron

   $cov(j)$ – function counts j-th neuron's coverage, which is the number of data referred to this neuron

   We can say, that this is the measure describes how often is such rule, how good quality , robust it is.

$$\begin{aligned} R_c(C(w_1), C(w_2)) &= \\ &= max\left(p_{zero}, R(C(w_1), C(w_2)) \times \frac{cov(win)}{\sum_j cov(j)}\right) \end{aligned} \tag{12}$$

   here:

   $C(w)$ – the class of word w

   w_1, w_2 – considered words

   $R(.)$ –result (thresholding function) function, that checks if cluster has enough high coverage:

$$R(C(w_1), C(w_2)) = \begin{cases} 1 & \text{if } cov(j) > covThreshold \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

$covThreshold$ – coverage threshold makes that clusters with enough high coverage are considered as positive rules.

2. *Distance from the center of the cluster* - may tell how the input $\mathbf{x}$ is close to represented by this cluster rule. The output $\mathbf{y}_{win}$ is reverse to such distance (see equations (6), (8), (9)). The language model result can be calculated in similar way as above:

$$\begin{aligned} R_d(C(w_1), C(w_2)) = max\big(&p_{zero}, \\ &R(C(w_1), C(w_2)) \times \mathbf{y}_{win}\left(C(w_1), C(w_2)\right)\big) \end{aligned} \tag{14}$$

3. *Threshold function.* Such solution will utilize only thresholding function $R(C(w_1), C(w_2))$ (eq. (13)).

$$R_t(C(w_1), C(w_2)) = max\big(p_{zero}, R\left(C(w_1), C(w_2)\right)\big) \tag{15}$$

As a simplest function it can help in consideration about result function. We will do the comparison if such a function should be flat, or should depend on cluster properties - eq. (12),(14).

One may noticed that we look for similarities to class based probabilistic model. To remind such a model [32] let express word probability by class of words probability. N-gram class based model:

$$P\left(w_k|w_{k-1}, \ldots, w_{k-n+1}\right) = P\left(w_k|C(w_k)\right) \times P\left(C(w_k)|C(w_{k-1}), \ldots, C_(w_{k-n+1})\right) \tag{16}$$

Here:

$P\left(C(w_k)|C(w_{k-1}, \ldots, C(w_{k-n+1})\right)$ is a probability of current word class given classes $C(w_{k-1}), \ldots, C_(w_{k-n+1})$ to which belongs previous words respectively $w_{k-1}, \ldots, w_{k-n+1}$.

$P\left(w_k|C(w_k)\right)$ is probability of occurience word w_k in all the occuriences words from class $C(w_k)$. It also mean probability of of word w_k occurience, knowing that this word belong to class $C(w_k)$.

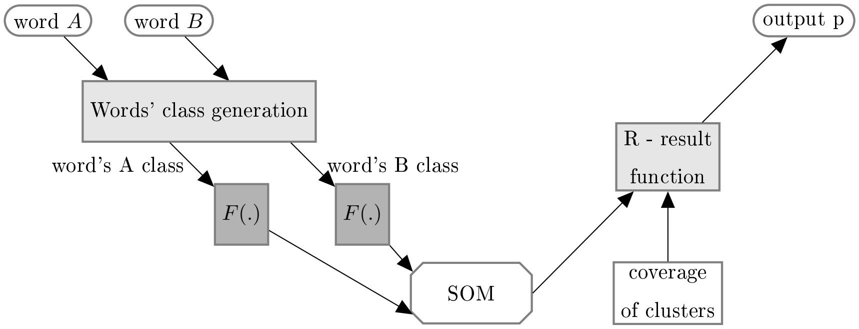


Figure 3: The calculation of resulting value p given two words

The classes of words are defined in this work from their linguistic properties.

The probability values set is continuous: $[0, 1]$, whereas using expression (12) here the threshold th is introduced. It is expressed in normalised probability and come from coverage threshold $covThreshold$ (normalisation will be explained below). It gives the probability range $P_{zeronorm} \cup (th, 1]$. $P_{zeronorm}$ refers to p_{zero} normalized. This consideration stays in the opposition to explanation from beginning of this section, which state that continuous value set would be suitable to the language model. We should notice that when we look for associations of words' in flexible word order language we will have occuriences of such words' pairs, which are not associated. This accidetal pairs should give quite wide distribution according to words and their linguistic propersties. Treating the as an information noise, we can expect that correct words' associations will get more frequent occuriences than them. Adjusted $covThreshold$ can differ this cases. In experiments will also be shown the need of such treshold. The other soultion instead step function R might be function with continuous, soft changes of its values (but still with slope around $covThreshold$ value).

The model proposed so far is presented on fig. 3. It will be expanded to new fascilities, which finally will gives complete language model.

6.2. Probability normalisation

For the evaluation purpose of speech hypothesis (formula (2)), we need probability. However the value p which calculation was described in previous section,still don't follow the definition of probability. There is need to normalise it.

Calculated values p are normalized by the sum of all measures p for the same class of word w_1, so the conditional probability:

$$P(A|B) = \frac{p(A|B)}{\sum_{A \in \Omega} p(A|B)} \tag{17}$$

which cause the need the summation of language model resulting function for all possible classes $C(w_2) \in \mathcal{C}$ which stay at position of word w_2 given class $C(w_1)$ of word w_1. $\mathcal{C}$ is the set of all classes.

Word probability of bigram class based language model, similar to (16):

$$P(w_1|w_2) = P(w_1|C(w_1)) \times P(C(w_1)|C(w_2)) \tag{18}$$

Probability of word w_1 occurience when this word belongs to class $C(w_1)$ we assume as a flat distribution for a class $C(w_1)$:

$$P(w_k|G(w_k)) = \frac{1}{n(G(w_k))} \tag{19}$$

We can rewrite the formulas (26),(12),(14) together, when take into account:

1. Quantity of the cluster:

$$P(C(w_1)|C(w_2)) = \begin{cases} \frac{max\left(p_{zero}, r_c(C(w_1), C(w_2))\right)}{S_c} & \text{if } S_c > 0 \\ \frac{1}{\|\mathcal{C}\|} & \text{if } S_c = 0 \end{cases} \tag{20}$$

   where:

$$r_c(c_1, c_2) = R(c_1, c_2) \times \frac{cov(win)}{\sum_j cov(j)} \tag{21}$$

$$S_c = \sum_{c \in \mathcal{C}} max\left(p_{zero}, r_c(c, C(w_2))\right) \tag{22}$$

2. Distance from the center of the cluster:

$$P(C(w_1)|C(w_2)) = \begin{cases} \frac{max\left(p_{zero}, r_d(C(w_1), C(w_2))\right)}{S_d} & \text{if } S_d > 0 \\ \frac{1}{\|\mathcal{C}\|} & \text{if } S_d = 0 \end{cases} \tag{23}$$

where:

$$r_d(c_1, c_2) = R(c_1, c_2) \times \mathbf{y}_{win}(c_1, c_2) \tag{24}$$

$$S_d = \sum_{c \in \mathcal{C}} max(p_{zero}, r_d(c, C(w_2))) \tag{25}$$

3. Threshold function:

$$R_t(C(w_1), C(w_2)) = max\big(p_{zero}, R\left(C(w_1), C(w_2)\right)\big) \tag{26}$$

$$P(C(w_1)|C(w_2)) = \begin{cases} \frac{max\left(p_{zero}, R(C(w_1), C(w_2))\right)}{S_d} & \text{if } S_d > 0 \\ \frac{1}{\|\mathcal{C}\|} & \text{if } S_d = 0 \end{cases} \tag{27}$$

where:

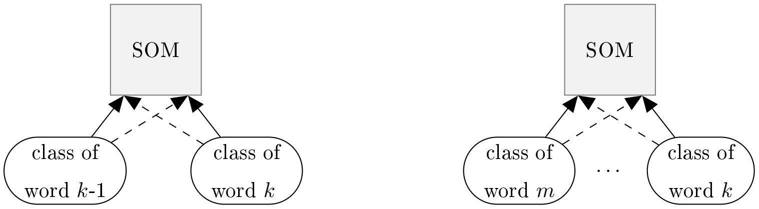
$$S_d = \sum_{c \in \mathcal{C}} max(p_{zero}, R(c, C(w_2))) \tag{28}$$

Such calculated result, a it was mentioned, can be taken as a word class conditional probability. The discussed part of the language model module can be seen on fig. 5. We can notice here folowing blocks: core SOM network, coding function $F(.)$, result function $R(.)$, and probability normalisation module. The inputs and output will be connected to other modules, and presented part of language model will be called as *Two words' classes LM SOM module.*

6.3. Network architecture

The way how information about word classes are taken from input, determine scope, kind of relations between words considered by language module.

PairSOM - the network inputs constitute concatenation of POS coding of two adjacent words. During training they are put in a forward order, and later, in a reverse order (Fig. 4a). It results in the fact that associations between such words can be learned without preserving their order. During the operation of such a model, there is a need to put the information about POS only once (in the forward order). It allows to detect associations between classes of such words independently of their order. This network has no ability of finding relationships between phrases or between non-adjacent words.



(a) the *PairSOM* network

(b) the *BinaryRelationSOM* network

Figure 4: The *PairSOM* and the *BinaryRelationSOM* language models

BinaryRelationSOM - the idea behind this network is to extend the scope of relationships between words (Fig. 4). One can put the information about POS from the all possible words within the given, limited context. The input of SOM network is concatenation of POS coding that comes from the k-th and m-th words ($k > m$), where the context length $n > k - m$. Next the order of these two words is reverted. The above procedure is iterated over all possible m. During the working phase of such a network given words are put in similar way, but only in forward order. Why the length of the context is limited? As we can expect the connections between more distant words should be less frequent. The values returned by network for each m are taken for further calculation. Probabilities, that were calculated (according to sections 6.1, 6.2) are multiplied:

$$P(w_k|W_{k-n+1}^{k-1}) = \prod_{k-n+1 \leq m < k} P(w_k|w_m) \text{ where } n = min(N, k) \tag{29}$$

When $n = 2$ this network is equivalent fo *PairSOM*. The presented network can find relationships between words, that do not has to be adjacent. They has to be within the maximal context length n. Such a model cannot find relationships between phrases, but can be necessary for speech recognition support. The work of this model could be compared with shallow parsing, since the result is not full parsing tree. The relationships' structure is flat and some of them may be not found. In comparison to consideration the full length of utterance (from beginning till current word) parsing the limited context brings smaller

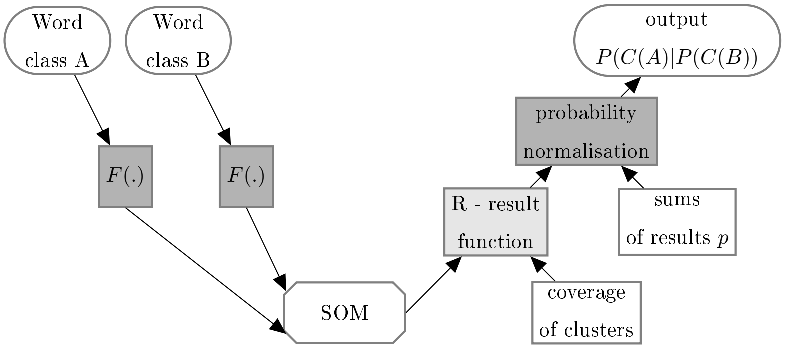


Figure 5: Two word's classes LM SOM module

computational complexity. This solution can be compared to Structured Language Model (for example [26]), where parsing is performed inside context of limited length.

To complete the discusion about network architecture and probability calculation we need should discuss which words should be send to normalisation module. *PairSOM* network will get words' pair only once, in forward order. Then in determination $P(w_k|w_{k-1})$ the normalisation will be done fo given w_{k-1}. In the same way for *BinaryRelationSOM* each probability $P(w_k|w_m)$ will need normalisation for given respective word w_m.

Modules, which were described there are presented on fig. 6. The Multiplexer module is responsible for choosing positions of two words in each step, to take their classes. Multiplication module gives the product of *Two words' clasess LM SOM module* results for the same current word w_k according to formula (29). Next the class probability is multiplied by word probability given its class. The word class generator gives all possible classes for each word. It is because of ambiguity in languge. The class of given word can be establilshed when it is connected to other words.

6.4. Word class determination

Solution known as morphosyntactic desambiguation rely on determination of linguistic properfties of word depend on its context. From given words there

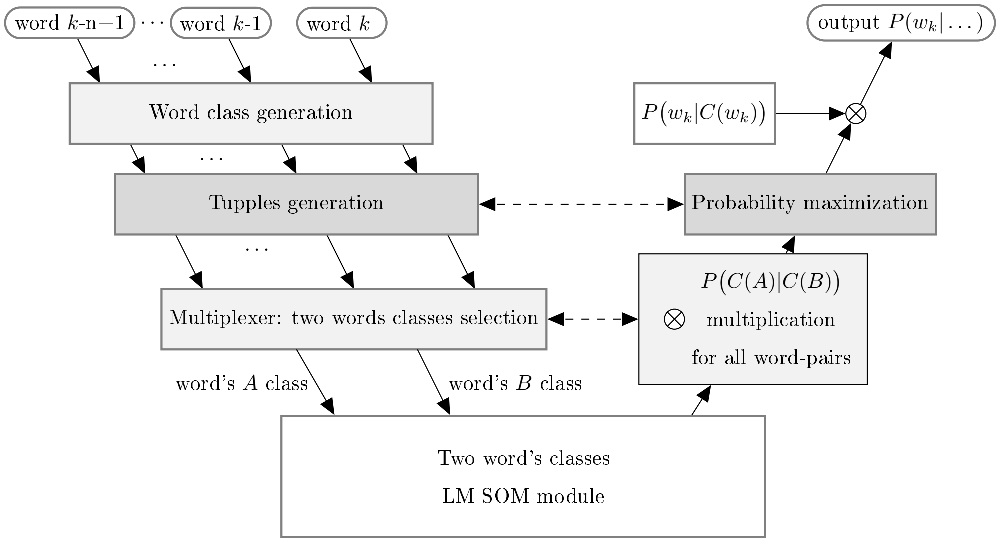


Figure 6: Full LM SOM module

are determined all possible tuple of classes. The lists L_i of all possible classes for each word w_i are generated. Then all possible tuples c are created, which has class $c_m \in L_m$ on element m. Here is proposed approach that for each such tuple the word probability will be calculated by already presented modules of language model. Next there is determned which tuple gives the highest probability. Having know given context usuch tuple refer to the most probable gramatical interpretation of given words.It will be choosen as a best one, and its probability will be returned as a final result of the model.

Described procedure can be writen as:

$$lm = max_{c=(c_1,c_2,\dots,c_i)\in(L1,L2,\dots,L_i)} LM(c) \tag{30}$$

which mean maximisation of word probabilities by finding word classes,which come from given lists L_i.

The drawback of this solution is limited context, that might be not enough lenght for all considered words.

7. Bibliography styles

There are various bibliography styles available. You can select the style of your choice in the preamble of this document. These styles are Elsevier styles based on standard styles like Harvard and Vancouver. Please use BibTEX to generate your bibliography and include DOIs whenever available.

Here are two sample references: [11].

References

[1] J. Benesty, M. M. Sondhi, Y. A. Huang, Springer Handbook of Speech Processing, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[2] F. Jelinek, Statistical methods for speech recognition, MIT Press, Cambridge, MA, USA, 1997.

[3] S. F. Chen, J. Goodman, An empirical study of smoothing techniques for language modeling, Computer Speech and Language 13 (4) (1999) 359 – 393, last visited 05.2013. `doi:10.1006/csla.1999.0128`. URL `http://www.sciencedirect.com/science/article/pii/S0885230899901286`

[4] S. Jerzy, Żołnierek Andrzej, Pipelined language model construction for polish speech recognition.

[5] Łukasz Brocki, D. Koržinek, K. Marasek, Telephony Based Voice Portal for a University, Speech and Language Technology 11 (2008) 55 – 58.

[6] D. Koržinek, Łukasz Brocki, Grammar Based Automatic Speech Recognition System for the Polish Language, in: R. Jabłoński, M. Turkowski, R. Szewczyk (Eds.), Recent Advances in Mechatronics, Springer Berlin Heidelberg, 2007, pp. 87–91. `doi:10.1007/978-3-540-73956-2_18`.

[7] C. Pollard, I. Sag, Head-Driven Phrase Structure Grammar, The University of Chicago Press/CSLI Publications, Chicago, IL, 1994.

[8] A. Przepiórkowski, A. Kupść, M. Marciniak, A. Mykowiecka, Formal description of Polish language - Theory and implementation [in Polish: Formalny opis języka polskiego - Teoria i implementacja], Problemy współczesnej nauki. Teoria i zastosowania. Inżynieria Lingwistyczna., Akademicka Oficyna Wydawnicza EXIT, Warszawa, 2002.

[9] L. Gajecki, R. Tadeusiewicz, Modeling of Polish language for Large Vocabulary Continuous Speech Recognition, in: G. Demenko, K. Jassem, S. Szpakowicz (Eds.), Speech and Language Technology, Vol. 11/2008, Polish Phonetic Association, Poznań, 2009, pp. 55 – 58.

[10] A. Przepiórkowski, Shallow processing of Polish language [in Polish: Powierzchniowe przetwarzanie języka polskiego], Problemy współczesnej nauki. Teoria i zastosowania. Inżynieria Lingwistyczna., Akademicka Oficyna Wydawnicza EXIT, Warszawa, 2008.

[11] L. Gajecki, Modelowanie jezyka naturalnego (polskiego) dla potrzeb budowy systemu rozpoznawania mowy klasy lvcsr (in polish: Natural language modeling of polish language for purposes of construction large vocabulary continuous speech recognition system), Ph.D. thesis, AGH – University of Science and Technology, Kraków, Poland, last visited 09.2015 (2013).
URL http://winntbg.bg.agh.edu.pl/rozprawy2/10679/full10679.pdf

[12] W. Chou, B. Juang (Eds.), Pattern Recognition in speech and language processing, Boca Raton : CRC Press, 2003.

[13] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. A. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, P. Woodland, HTK Book, Cambridge University, Engineering Department, 2009, last visited 04.2012.
URL http://htk.eng.cam.ac.uk

[14] J. Duchateau, Hmm based acoustic modeling in large vocabulary speech recognition, Ph.D. thesis, Katholieke Universiteit Leuven, Belgium, last visited 01.2013 (1998).
URL `http://www.esat.kuleuven.be/psi/spraak/`

[15] M. Szymański, J. Ogórkiewicz, M. Lange, K. Klessa, S. Grocholewski, G. Demenko, First evaluation of Polish LVCSR acoustic models obtained from the JURISDIC database, Speech and Language Technology 11 (2008) 39–46.

[16] A. Acero, Wang, Wang, A semantically structured language model.

[17] H. Erdogan, R. Sarikaya, S. Chen, Y. Gao, M. Picheny, Using semantic analysis to improve speech recognition performance, Computer Speech and Language 19 (3).

[18] L. ten Bosch, L. Boves, H. V. Hamme, R. K. Moore, A Computational Model of Language Acquisition: the Emergence of Words, Fundamenta Informaticae 90 (3) (2009) 229–249. `doi:10.3233/FI-2009-0016`.

[19] W. Xu, A. Rudnicky, Can Artificial Neural Networks Learn Language Models?, in: Proceedings of ICSLP 2000, Beijing, China, 2000.

[20] Y. Bengio, H. Schwenk, J.-S. Senécal, F. Morin, J.-L. Gauvain, Neural probabilistic language models, in: D. E. Holmes, L. C. Jain (Eds.), Innovations in Machine Learning - Theory and Applications, Studies in Fuzziness and Soft Computing, Springer-Verlag, Berlin, Heidelberg, 2006, pp. 137–186.

[21] M. Castro, F. Prat, New directions in connectionist language modeling, in: J. Mira, J. Álvarez (Eds.), Computational Methods in Neural Modeling, Vol. 2686 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2003, pp. 598–605. `doi:10.1007/3-540-44868-3_76`.
URL `http://dx.doi.org/10.1007/3-540-44868-3_76`

[22] Łukasz Brocki, Koneksjonistyczny model języka w systemach rozpozanwania mowy, Ph.D. thesis, Warszawa (2010).

[23] F. Morin, B. Yoshua, Hierarchical probabilistic neural network language model.

[24] C. Chelba, F. Jelinek, Structured language modeling, Computer Speech and Language 14 (4) (2000) 283 – 332, last visited 05.2013. doi:10.1006/csla.2000.0147.
URL http://www.sciencedirect.com/science/article/pii/S0885230800901475

[25] D. H. V. Uytsel, D. V. Compernolle, Language modeling with probabilistic left corner parsing, Computer Speech and Language 19 (2) (2005) 171 – 204. doi:http://dx.doi.org/10.1016/j.csl.2004.05.009.
URL http://www.sciencedirect.com/science/article/pii/S0885230804000221

[26] A. Emami, F. Jelinek, A Neural Syntactic Language Model, Machine Learning 60 (1-3) (2005) 195–227, last visited 07.2013. doi:10.1007/s10994-005-0916-y.
URL http://dx.doi.org/10.1007/s10994-005-0916-y

[27] H. Ritter, T. Kohonen, Self-organizing semantic maps, Biological Cybernetics 61 (4) (1989) 241–254. doi:10.1007/BF00203171.
URL http://dx.doi.org/10.1007/BF00203171

[28] L. Gajecki, Architectures of neural networks applied for LVCSR language modeling, Neurocomputing 133 (2014) 46 – 53.
URL http://dx.doi.org/10.1016/j.neucom.2013.11.033

[29] T. Kohonen, Self-Organizing Maps, Springer, 2001.

[30] R. Tadeusiewicz, New trends in neurocybernetics, Computer Methods in Materials Science 10 (2010) 1–7, last visited 08.2013.
URL www.cmms.agh.edu.pl

[31] L. Gajecki, R. Tadeusiewicz, Language modeling using SOM network, in: Z. Vetulani (Ed.), Procedings 5th Language and Technology Conference, Fundacja UAM, Poznań, Poland, 2011, pp. 216–220.

[32] P. F. Brown, P. V. de Souza, R. L. Mercer, V. J. D. Pietra, J. C. Lai, Class-Based n-gram Models of Natural Language, Computational Linguistics 18 (1992) 467–479.