Gupta, PA.; Branchs, R.; Rosso, P. (2016). Squeezing Bottlenecks: Exploring the Limits of Autoencoder Semantic Representation Capabilities. Neurocomputing. 175:1001-1008. doi:10.1016/j.neucom.2015.06.091.

Squeezing bottlenecks: exploring the limits of autoencoder semantic representation capabilities

Parth Gupta[a,*], Rafael E. Banchs[b], Paolo Rosso[a]

[a] *PRHLT Research Center, Universitat Politècnica de València, Spain*
[b] *Institute for Infocomm Research, Singapore*

Abstract

We present a comprehensive study on the use of autoencoders for modelling text data, in which (differently from previous studies) we focus our attention on the following issues: *i)* we explore the suitability of two different models bDA and rsDA for constructing deep autoencoders for text data at the sentence level; *ii)* we propose and evaluate two novel metrics for better assessing the text-reconstruction capabilities of autoencoders; *iii)* we propose an automatic method to find the critical bottleneck dimensionality for text language representations (below which structural information is lost); and *iv)* we conduct a comparative evaluation across different languages, exploring the regions of critical bottleneck dimensionality and its relationship to language perplexity.

1. Introduction

One of the major hurdles in comparing text in vector space models (VSM) is to deal with problems like *synonymy* and *polysymy*. Usually in vector space, the documents are composed of thousands of dimensions. In addition to high computational complexity, many meaningful associations between terms are shadowed by large dimensions. There are models which try to solve this problem *e.g.* pseudo relevance feedback (PRF) and explicit semantic analysis (ESA) (Xu and Croft, 1996; Gabrilovich and Markovitch, 2007). Other category of attempts to solve this problem comprise of dimensionality reduction techniques.

*Corresponding author
Email addresses: pgupta@dsic.upv.es (Parth Gupta), rembanchs@i2r.a-star.edu.sg (Rafael E. Banchs), prosso@dsic.upv.es (Paolo Rosso)

The goal of dimensionality reduction techniques is to transform high dimensional data ($\mathbb{R}^n$) into a much lower dimension representation ($\mathbb{R}^m$) pertaining the inherent structure of the original data where $m << n$. One such widely used approach is latent semantic indexing (LSI) which extracts a low rank approximation of a term-document matrix by means of principal component analysis (PCA) (Deerwester et al., 1990). There are some advanced approaches like probabilistic latent semantic analysis (PLSA) and latent dirichlet allocation (LDA) which observe the distribution of latent topics for the given documents (Hofmann, 1999; Blei et al., 2003).

Dimensionality reduction techniques are also prominent while estimating the similarity between text across languages. Associations of terms and documents across languages in such techniques are learnt by means of parallel or comparable text (Nie et al., 1999; Banchs and Kaltenbrunner, 2008; Platt et al., 2010).

Dimensionality reduction techniques can broadly be categorised in two classes: linear and non-linear. Usually, non-linear techniques can find more compact representations of the data compared to their linear counterparts (Hinton and Salakhutdinov, 2006). If there exists statistical dependence among the principal components of PCA, or principal components have non-linear dependencies, PCA would require a larger dimensionality to properly represent the data when compared to non-linear techniques.

On the other hand, although non-linear projection methods such as multidimensional scaling (MDS) give a way to obtain much better representations for mono and cross-language similarity estimation, it is a transductive method (Cox and Cox, 2001; Banchs and Kaltenbrunner, 2008). It means MDS does not provide an operator to project the unseen data into the target low dimensional space like the resulting projection matrix in the case of PCA.

Lately, dimensionality reduction techniques based on deep-learning have become very popular, especially deep autoencoders (DA). Deep autoencoders can extract highly useful and compact features from the structural information of the data. Deep autoencoders have proven to be very effective in learning reduced space representations of the data for similarity estimation, *i.e.* similar documents tend to have similar abstract representations (Hinton and Salakhutdinov, 2006; Salakhutdinov and Hinton, 2009a). Deep learning is inspired by biological studies which state the brain has a deep architecture. Despite their high suitability to the task, deep-learning did not find much audience because of convergence issues until Hinton and Salakhutdinov Hinton and Salakhutdinov (2006) gave a way to initialise the network parameters in a good region for finding optimal solutions.

Although deep learning techniques are in vogue, there still exist some impor-

tant open questions. In most of the studies involving the use of these techniques for dimensionality reduction, the qualitative analysis of projections is never presented. This makes the assessment of the reliability of learning very difficult. Typically, the reliability of the autoencoder is estimated based on its reconstruction capability.

The first objective of this work is to propose a novel framework for evaluating the quality of the dimensionality reduction task based on the merits of the application under consideration: the representation of text data in low dimensional spaces.

Concretely, our proposed framework is comprised of two metrics, *structure preservation index (SPI)* and *similarity accumulation index (SAI)*, which capture different aspects of the autoencoder's reconstruction capability like the structural distortion and similarities among the reconstructed vectors. In this way, our proposed framework gives better insight of the autoencoder performance allowing for conducting better error analysis and evaluation, and, as explained below, these metrics also provides a better means for estimating the adequate size of critical bottleneck dimensions.

The second objective of this work is to conduct a comparative evaluation across different languages of the dimensionality reduction capabilities of deep autoencoders. With this empirical evaluation we aim at shedding some light regarding the adequacy of reducing different languages to a common bottleneck dimension, which is a common practice in the field.

We carry out the experiments of dimensionality reduction of text at sentence level and assess the suitability of two types of deep autoencoders. We report some interesting findings at the architectural level of the specific problem of modelling text at the sentence level.

The rest of the paper is structured as follows. A brief introduction to deep autoencoders is given in Section 2. Section 3 gives details about the analysis framework of the autoencoder learning, experiments and results. The discussion on critical bottleneck dimensionality and an automatic way to estimate it is given in Section 4. In Section 5, we attempt to relate the critical bottleneck dimensionality for a particular language to its perplexity. Finally, we present the conclusions and future directions of this work in Section 6.

2. Models

In this section we describe the models we have considered for performing dimensionality reduction of text data. First, we provide a brief introduction to

autoencoders. Then, in sub-section 2.1, we present the binary deep autoencoder model (bDA); and, in sub-section 2.2, we describe the replicated softmax deep autoencoder (rsDA). Finally, in sub-section 2.3, we discuss the training procedure in detail.

Both of the considered models differ in the way they model the text data. While the binary deep autoencoder models the presence of the term into the document (*binary*), the replicated softmax deep autoencoder directly models the count of the term (i.e., *term frequency*) in the document.

The autoencoder is indeed a network which tries to learn an approximation of the identity function so as the output is similar to input. The input and output dimensions of the network are the same (n). The autoencoder approximates the identity function in two steps: *i)* reduction, and *ii)* reconstruction. The reduction step takes the input $x \in \mathbb{R}^n$ and maps it to $y \in \mathbb{R}^m$ where $m < n$ which can be seen as a function $y = g(x)$ with $g : \mathbb{R}^n \rightarrow \mathbb{R}^m$. On the other hand, the reconstruction step takes the output of the reduction step y and maps it to $\hat{x} \in \mathbb{R}^n$ in such a way $\hat{x} \approx x$ which is considered as a $\hat{x} = f(y)$ with function $f : \mathbb{R}^m \rightarrow \mathbb{R}^n$. The full autoencoder can be seen as $f(g(x)) \approx x$.

In a neural network based implementation of the autoencoder, the visible layer corresponds to the input x and the hidden layer corresponds to y. There are two variants of autoencoders: *i)* with a single hidden layer, and *ii)* with multiple hidden layers. If there is only one single hidden layer, the optimal solution remains the PCA projection even with the added non-linearities in the hidden layer (Bourlard and Kamp, 1988). The PCA limitations are overcome by stacking multiple encoders, constituting what is called a deep architecture. This deep construction is what leads to a truly non-linear and powerful reduced space representation (Hinton and Salakhutdinov, 2006). The deep architecture is constituted by stacking multiple restricted boltzmann machines (RBM) on top of each other as shown in Fig. 1.

Each RBM is a two-layer bipartite network with a visible layer ($\mathbf{v}$) and a hidden layer ($\mathbf{h}$). Both layers are connected through symmetric weights ($\mathbf{w}$). Usually the hidden units correspond to *latent* variables. It is very easy to sample the data from visible to hidden layer and vice-versa. The two models we consider here, bDA and rsDA, primarily differ in the bottom-most RBM, i.e. the way they model the input data. In a nutshell, in the case of bDA, the bottom-most RBM is a standard RBM with stochastic binary (visible and hidden) layers; while, in the case of rsDA, the bottom-most RBM is based on the replicated softmax model (RSM) (Salakhutdinov and Hinton, 2009b).

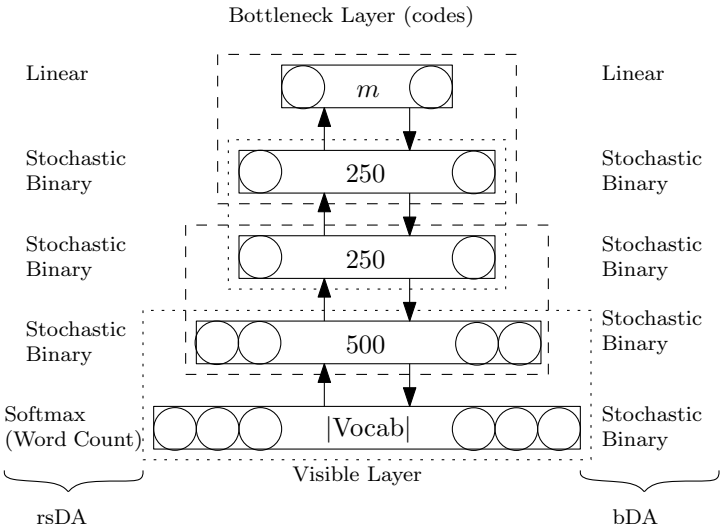

Figure 1: Architecture of the deep autoencoders. The binary and replicated softmax deep autoencoders are denoted as bDA and rsDA. |Vocab| is the size of vocabulary at the input layer.

2.1. Stochastic Binary RBM

Stochastic binary RBMs have both, visible and hidden, layers as stochastic binary with sigmoid non-linearity. Let visible units $\mathbf{v} \in \{0,1\}^n$ be binary bag-of-words representation of text documents and hidden units $\mathbf{h} \in \{0,1\}^m$ be the hidden latent variables. The energy of the state $\{\mathbf{v}, \mathbf{h}\}$ is as follows,

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{i=1}^{n} a_i v_i - \sum_{j=1}^{m} b_j h_j - \sum_{i,j} v_i h_j w_{ij} \tag{1}$$

where v_i, h_j are the binary states of visible unit i and hidden unit j, a_i, b_j are their biases and w_{ij} is the weight between them.

Then, it becomes easy to sample the data in both directions as shown below,

$$p(v_i = 1|\mathbf{h}) = \sigma(a_i + \sum_j h_j W_{ij}) \tag{2}$$

$$p(h_j = 1|\mathbf{v}) = \sigma(b_j + \sum_i v_i W_{ij}) \tag{3}$$

where $\sigma(x) = 1/(1 + \exp(-x))$ is the logistic sigmoid function.

2.2. Replicated Softmax RBM

The Replicated Softmax RBM is based on the Replicated Softmax Model (RSM) proposed by Salakhutdinov and Hinton Salakhutdinov and Hinton (2009b).

Let $\mathbf{v} \in \{1, \ldots, K\}^D$, where K is the vocabulary size, D is size of the document and let $\mathbf{h} \in \{0, 1\}^m$ be stochastic binary hidden latent variables. Considering a document with length D , the energy of the state $\{\mathbf{v}, \mathbf{h}\}$ is defined as,

$$E(\mathbf{v}, \mathbf{h}) = -\sum_{k=1}^{K} \hat{v}^k a^k - D\sum_{j=1}^{m} b_j h_j - \sum_{k,j} W_j^k h_j \hat{v}^k \tag{4}$$

where, $\hat{v} = \sum_{i=1}^{D} v_i^k$ denotes the count data for the k^{th} term.

In RSM, the visible layer is softmax with multinomial visible units which represents the probability distribution of the word-count. It is sampled D times by using multinomial sampling to recover the original word-count data. Another distinction of this model is scaling of the bias terms of the hidden layer which gives a way to handle the documents of different lengths. In this case, the visible and hidden units are updated as shown below,

$$p(v_i^k = 1|\mathbf{h}) = \frac{\exp(b_i^k + \sum_j h_j W_{ij}^k)}{\sum_{q=1}^{K} \exp(b_i^q + \sum_j h_j W_{ij}^q)} \tag{5}$$

$$p(h_j = 1|\mathbf{V}) = \sigma(a_j + \sum_{i=1}^{D}\sum_{k=1}^{K} v_i^k W_{ij}^k) \tag{6}$$

2.3. Training of Autoencoders

Autoencoders are typically trained in two steps: *i)* greedy layerwise pre-training, and *ii)* fine-tuning of the parameters to learn the identity approximation of the input data.

2.3.1. Pre-training

In this step, each RBM is trained greedily using contrastive divergence (CD) learning (Hinton, 2002). Here the RBMs are trained one by one starting from the bottom-most RBM. The bottom-most RBM directly takes the input data while the upper RBMs take the output $p(\mathbf{h}|\mathbf{v})$ of the RBM below which is already trained. We use the structure of the autoencoder 500-250-250-m as shown in Fig. 1. We train each RBM using CD_1 learning for 50 epoch where CD_1 refers to CD with 1 step of alternating Gibbs sampling (Hinton, 2002).

2.3.2. Fine-tuning

Once the RBMs are trained layer-wise, the autoencoder is unrolled as shown in Fig. 2. The stochastic binary activities of the feature layers is replaced by the real-valued probabilities and the input data is backpropagated through the network to fine-tune the parameters of the entire network. We calculate the cross-entropy error (e) between $\hat{x} = f(g(x))$ and x as shown below and backpropagate it through the entire network.

$$e = -\sum_{i=1}^{n}[x_i \log(\hat{x}_i) + (1 - x_i)\log(1 - \hat{x_i})] \tag{7}$$

In case of bDA the binary input data is used to calculate e. While for rsDA, the word-count input vectors are divided by the document length (D) to represent probability distribution which together with softmax output layer is used to calculate e.

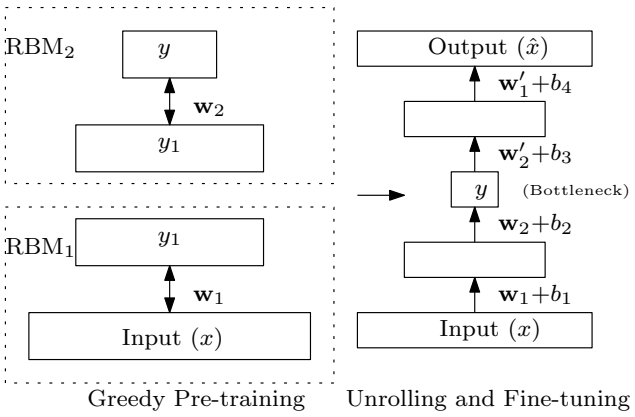

Figure 2: **Left panel**: pre-training the stacked RBMs where upper RBMs take output of the lower RBM. **Right panel**: After pre-training the structure is "unrolled" to create a multi-layer autoencoder which is fine-tuned by backpropagation to perform $\hat{x} \approx x$.

3. Qualitative Analysis and Metrics

In this section we describe the proposed metrics used for comparing the bDA and rsDA models. Subsequently, we present the comparative analysis of the two models.

The quality of the projections and the sufficiency of dimension m are measured by the autoencoder's reconstruction ability. Unfortunately, the mean squared

error between the input x and its reconstruction $\hat{x}$, referred as *reconstruction error*, is a poor measure to estimate it. It neither gives any details about the quality of the reconstructions in terms of text data representation nor the degree to which the structure of the data is preserved in the reconstruction space. Moreover, it is difficult to justify the adequacy of bottleneck dimension m by simply using the *reconstruction error*.

In literature, when autoencoders are used for dimensionality reduction for text data, most of the time, the quality of the algorithm is measured in terms of the accuracy of the end-task which may be text categorisation (Hinton and Salakhutdinov, 2006), information retrieval (Salakhutdinov and Hinton, 2009a), topic modeling (Salakhutdinov and Hinton, 2009b), term modeling across scripts (Gupta et al., 2014) or sentiment prediction (Socher et al., 2011). A shortcoming of this approach is that there is no way to estimate the full potential, or the upper bound, of the algorithm performance. On the other hand, in the case of poor results, it becomes tougher to decide whether the training was proper or not.

As already mentioned before, in this work we propose two new metrics: *i) structure preservation index (SPI)*, and *ii) similarity accumulation index (SAI)*, which are intended to capture different aspects of the autoencoder's reconstruction capability, like the structural distortion and semantic similarity of the reconstructed vectors with respect to the original ones. Considering these two metrics, along with the *reconstruction error*, allows for a much better assessment of confidence regarding the quality of the network training process and its performance.

Structure Preservation Index (SPI):. Consider the input data as X where each row x_i corresponds to the vector space representation of the i^{th} document and $\hat{X}$ is its corresponding reconstruction. X and $\hat{X}$ are pxn matrices where p is the total number of documents in the input data and n is the vocabulary size. Compute matrix D for X such that D_{ij} is the cosine similarity score between i^{th} and j^{th} rows of X. Similarly calculate $\hat{D}$ for $\hat{X}$. D and $\hat{D}$ can be seen as dissimilarity matrices of the original data and its reconstruction, respectively, where $D_{ij} = \hat{D}_{ij} = 1, \forall i = j$. The SPI is calculated as follows,

$$\text{SPI} = \frac{1}{p^2} \sum_{ij} ||D_{ij} - \hat{D}_{ij}||^2 \tag{8}$$

Notice that according to this definition, SPI captures the structural distortion incurred by the $f(g(X))$ process. Ideally, SPI should be zero.

Similarity Accumulation Index (SAI):. Different from SPI, which assesses structural distortion, SAI attempts to capture the quality of the reconstructed vectors by measuring the cosine similarity between each original vector and its reconstructed version. Indeed, this verifies the preservation of the relative strength of the vector-dimensions in the reconstruction.

SAI is computed by the normalised accumulation of cosine similarities between each input document and its reconstruction. Ideally, SAI should be one.

$$\text{SAI} = \frac{1}{p} \sum_{i=1}^{p} \text{cosine}(x_i, \hat{x}_i) \tag{9}$$

3.1. Comparative Evaluation of Models

We carried out an experiment of dimensionality reduction for text sentences, where data sparseness plays a more critical role than in the case of full documents (dimensionality reduction applied to full documents is the case that has been mostly explored in the literature).

In this study we aim at applying autoencoder techniques at the level of sentences to open its way for sentence-centered applications, such as machine translation, text summarization and automatic dialogue response.

For this experiment, we used the Bible dataset, which contains 25122 training and 995 test sentences. All sentences were processed by a term-pipeline of stopword-removal and stemming which is referred as $\textbf{Vocab}_1$. After that we kept only those terms which were non-numeric, at least 3-characters long and appeared in at least 5 training sentences. We refer to this filtered vocabulary as $\textbf{Vocab}_2$. For English partition of the dataset, $\textbf{Vocab}_1$ and $\textbf{Vocab}_2$ are 8279 and 3100 respectively.

Next, we present the results for English using both models, bDA and rsDA, and present the qualitative analysis of the reconstructions with the help of the aforementioned metrics. We train both autoencoders with the structure 500-250-250-40 as described in Section 2.3. The results are presented in Table 1.

3.2. Analysis and Discussion

When operating in vector space, it is important to understand the amount of distortion incurred by the network on the structure of the data during the process of $f(g(x))$. The network uses the *reconstruction error* during the training to update parameters but it does not give much insight about the quality of the network.

Model	RC	SPI	SAI
rsDA (*pt*)	0.1192	0.7258	0.2132
rsDA (*bp*)	0.0834	0.0049	0.5768
bDA (*pt*)	8.0012	0.0712	0.3528
bDA (*bp*)	5.4829	0.0035	0.6667

Table 1: The performance of bDA and rsDA in terms of different metrics. **RC** denotes *reconstruction error* while *pt* and *bp* denote if the model is only pre-trained and fine-tuned after pre-training, respectively.

One more limitation of the *reconstruction error* is that it is not bounded and not comparable across different models $e.g.$ bDA and rsDA. The *reconstruction error* is calculated between the softmax output and the probability distribution of terms in case of rsDA hence it is not comparable to that of bDA (see Table 1).

The two proposed metrics, SPI and SAI are both bounded by [0,1] and comparable across the models. SPI gives the measure of how the similarity structure of sentences among each other is preserved in the reconstruction space which in turn gives a measure of trustworthiness of the network for similarity estimation. Although both models show descent performance in terms of SPI after backpropagation, bDA is 28.57% better than rsDA in terms of SPI.

It is also important to assess the similarity between each input vector and its corresponding reconstruction which is captured by SAI. In terms of SAI, bDA is 15.59% better than rsDA. This is better understood in Fig. 3, where it can be noticed that, in the case of rsDA, for more than half of the test samples cosine similarity with their reconstruction is ≤ 0.6. Although rsDA has been reported in the literature to better perform at the document level, our results demonstrate that bDA is a more suitable model to be used when using autoencoder representations at the sentence level. This can be explained by the fact that rsDA uses multinomial sampling to model the word-count, which happens not to be suitable at the sentence level for three reasons: *i)* most of the terms appear only once in the sentences, *ii)* sampling the distribution of terms D times is less reliable when D is quite small which is the case in sentences compared to full documents *iii)* the gradients at the output layer (softmax) are very small as they are calculated over probability distribution.

Finally, as argued by Erhan et al. Erhan et al. (2010), pre-training helps to initialise the network parameters in a region to find optimal solution. It can clearly be noticed that pre-training is necessary but itself is not enough to put aside backpropagation.

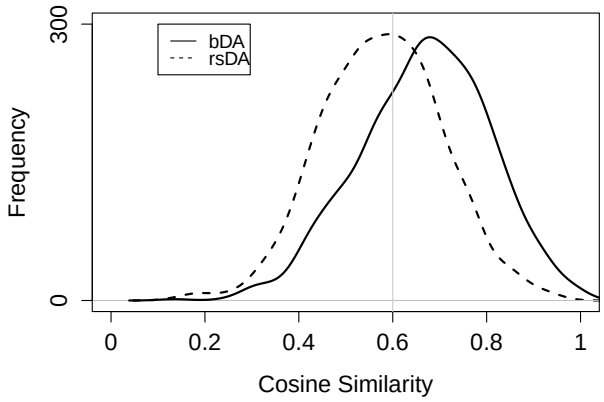

Figure 3: Histogram of cosine similarity between test samples and their reconstructions for bDA and rsDA.

4. Critical Bottleneck Dimensionality

In this section we present the analysis on the adequacy of the size of bottleneck layer. Later, we extend the analysis to multilingual scenario and describe the automatic method to estimate the critical bottleneck dimensionality for different languages.

The top-most hidden layer of an autoencoder is commonly referred to as the bottleneck layer. The reconstruction ability of the autoencoder is highly related to the size of the bottleneck layer, in the sense that the smaller the size of the bottleneck layer is, the higher the loss of information is.

The reduction step of autoencoders is also called *hashing*, and because similar sentences in the projected space are near to each other, this technique is also referred to as *semantic hashing*. It is important to choose a proper size of the bottleneck layer because of two reasons: *i)* a too large size may lead to redundant dimensions and high computational cost, and *ii)* a too small size might lead to high information loss.

The adequacy of the bottleneck dimension, which we refer to as critical bottleneck dimensionality here, is rarely addressed in the literature. In this section of the study, we present an analysis on the effects of choosing different sizes for the bottleneck layer, as well as we provide an empirical method to choose the critical bottleneck dimensionality properly.

4.1. Metric Selection

We squeeze the bottleneck layer of the autoencoder to identify whether there was a dimensionality region at which the *reconstruction error*, SPI and SAI metrics exhibit a clear change in its behaviour. Typically, this region is referred to as the "elbow region". We trained the autoencoder varying down the size of the bottleneck layer from 100 to 10 with step-sizes of 10. Fig. 4 shows the values of *reconstruction error*, SPI and SAI for different sizes of bottleneck layer. As

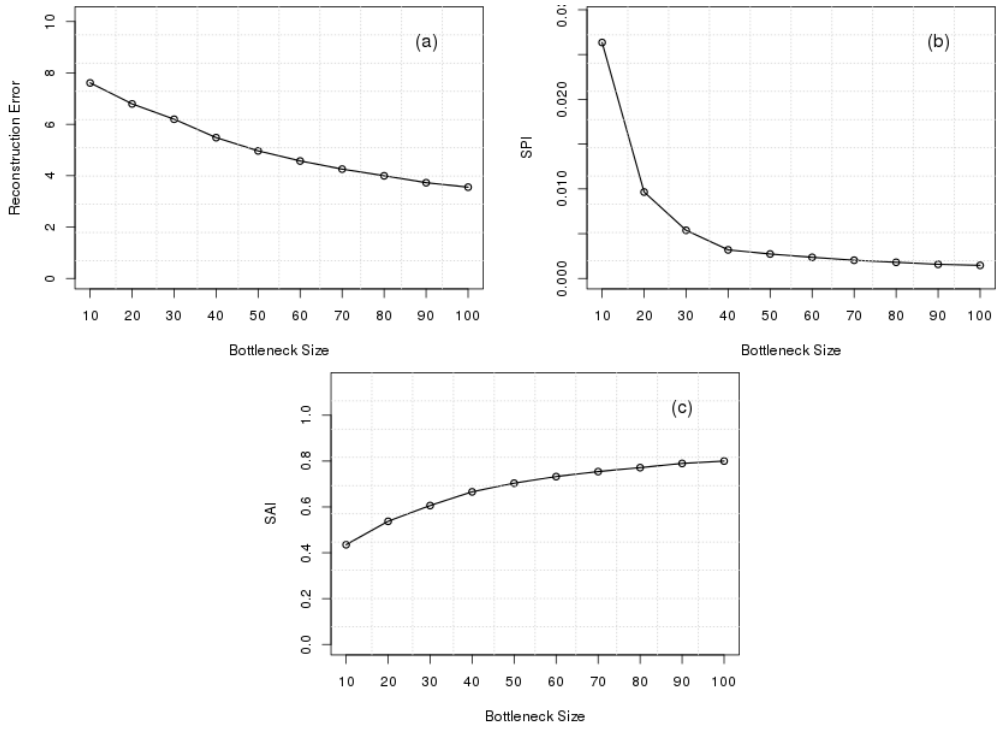

Figure 4: *Reconstruction error*, SPI and SAI metrics while squeezing the bottleneck layer from 100 to 10 are shown respectively in *(a)*, *(b)* and *(c)*.

it becomes evident from the figure, SPI is the metric exhibiting the clearest "elbow region" pattern, hence we will use this metric for determining the critical bottleneck dimensionality. Indeed, it can be noticed that both the *reconstruction error* and SAI show a quasi-linear behaviour with almost constant slope, while SPI clearly captures that below $m = 40$, the network starts losing the structural information within the data. This result shows that care must be taken to select a proper bottleneck dimension and it is important not to choose the bottleneck dimension below the point where SPI changes its behaviour.

4.2. Multilingual Analysis

Typically, in cross- and multi-language dimensionality reduction techniques, most often all the documents are projected to a common dimensionality abstract space regardless the language. Based on the analysis presented in the previous section regarding the critical bottleneck region, it becomes important to assess the following research question: *does a common dimension suit all the languages?*

To understand this phenomenon, we conduct a comparative study by considering different-language versions of the same English dataset already used in Sections 3.1 and 4.1. Due to language pre-processing capabilities, we restricted our study to 5 different and diverse languages: English (Indo-European/Germanic), Spanish (Indo-European/Italic), Russian (Indo-European/Balto-Slavic), Turkish (Turkic) and Arabic (Afro-Asiatic), for all of which we repeated the experiment described in Section 4.1. The statistics of the vocabulary sizes at these languages is depicted in Table 2. The fundamental idea behind this experiment is to see

Language	$\lvert\textbf{Vocab}_1\rvert$	$\lvert\textbf{Vocab}_2\rvert$
English (en)	8279	3100
Spanish (es)	9398	3581
Russian (ru)	18285	4504
Turkish (tk)	17087	4502
Arabic (ar)	18703	3012

Table 2: Statistics of the Bible dataset.

whether the same knowledge (parallel corpus) in different languages can be represented by a reduced dimensionality space of the same size. We anticipated that the critical bottleneck dimensionality of each language can be affected by different parameters like: its vocabulary size, its syntactic structure and its semantic complexity.

To identify the critical bottleneck dimensionality for each language, we calculated the percentage difference between the slopes connecting consecutive bottleneck sizes in the SPI curve. This captures the point in the "elbow region" at which the slope of the SPI curve is steepest. Consider three points in SPI plot: a_1, a_2 and a_3. Let s_1^2 and s_2^3 be the slopes of lines connecting $a_1 - a_2$ and $a_2 - a_3$, respectively. Then the percentage difference between s_1^2 and s_2^3 gives the steepness of the curve at point a_2. We calculate this figure for every point in the range and estimate the *critical dimensionality* at which the percentage difference is the largest. This method enables us to automatically find the adequate bottleneck dimension for a

particular language. The algorithmic implementation of this method is described in Fig. 5.

Method: Estimation of *critical* dimension
Input: A, B **Output**: C A = set of bottleneck dimensions B = set of SPI values, where b_i = SPI$(a_i) \in A$ C = set of steepness values at each point **for each** $a_{i-1}, a_i, a_{i+1} \in A$ **get** $b_{i-1}, b_i, b_{i+1} \in B$ **calc.** s_{i-1}^{i}, s_{i}^{i+1} where, s_{i-1}^{i} = slope((a_{i-1}, b_{i-1}), (a_i, b_i)) **calc.** c_i = % diff (s_{i-1}^{i},s_{i}^{i+1}) **add** c_i to C **end** **plot** C *critical dim.* = right-most large peak

Figure 5: Method to identify the *critical dimensionality* for the bottleneck layer for a particular language.

For providing a better graphical representation on how the critical bottleneck dimensionality is identified, Fig. 6 shows the the second derivative approximation of the SPI curve that is computed by the method for all the different languages under consideration. For some languages, there is a clear single peak where the SPI curve changes its behaviour drastically *e.g.* English, Spanish and Turkish. On the other hand, for some other languages, there exist multiple large peaks *e.g.* Russian and Arabic. In these latter cases, the right-most large peak is the one considered indicative of the critical bottleneck dimensionality. This is mainly because further below that point the network drastically loses the capacity for recovering the original data structure information.

Finally, it should be mentioned that the *critical bottleneck dimensionality* might not be easily spotted directly from the slope of SPI curve, but plotting the percentage difference, which approximates the SPI's second derivative, clearly captures it. It is evident from the results presented in this section that different language exhibit different *critical bottleneck dimensionalities*. This provides a much more

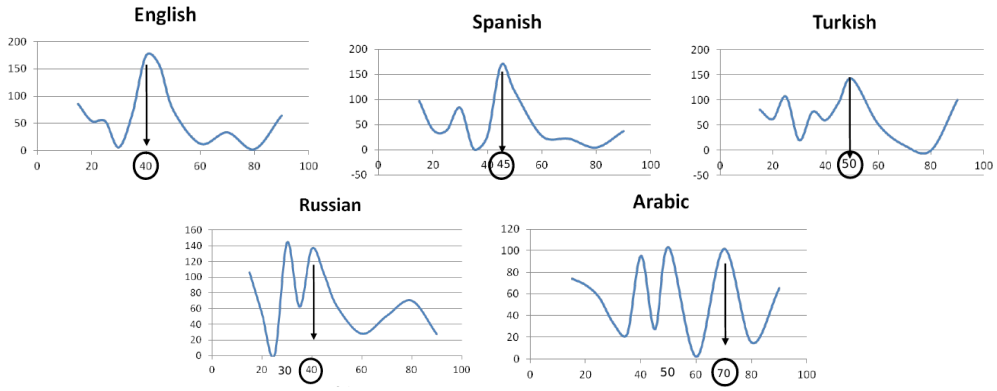

Figure 6: The percentage difference in slope of the SPI curve at each dimension.

principled criterion for the selection of the target dimensionalities in cross- and multi-language applications that use dimensionality reduction techniques.

5. Critical Dimensionality and Perplexity

It has been discussed that the neocortex of the brain works in multiple layers where each layer captures some specific type of information (Quartz and Sejnowski, 1997; Utgoff and Stracuzzi, 2002). This presents a strong analogy to the computational deep learning framework. Inspired on this evidence, we anticipated that the critical bottleneck dimensionality of each language can be affected by their different structural and semantic characteristics.

As an additional empirical analysis, we used the word-level perplexities of each considered language as a proxy to its structural and semantic complexity, and we evaluated whether such a proxy correlates with the critical bottleneck dimensionalities obtained in the previous section.

Perplexity is often used as a metric for evaluating the quality of a language model. A word-level perplexity of value V indicates that the considered model found V alternatives for each term; therefore, the better a model is, the lower the resulting perplexity. In the limit, the lowest perplexity achievable by a language model indicates the actual information content (entropy) of the given language (Brown et al., 1992).

In order to establish whether the language information content and its critical bottleneck dimensionality are related to each other, we calculated the Pearson's correlation coefficient between the word-level perplexity estimated with a trigram model and the critical bottleneck dimensionalities obtained in the previous sec-

tion. Table 3 presents the obtained perplexities for both, token and stem based, trigram models along with the critical bottleneck dimensionalities for each of the five languages under consideration; and Table 4 presents the resulting correlation coefficients and their corresponding p-values. As observed from Table 4, although

Lang.	**Crit. Dim.**	**PPL-T**	**PPL-S**
en	40	64.0018	59.6428
es	45	113.075	89.4268
tk	50	322.315	177.117
ru	40	218.634	159.588
ar	70	741.115	296.663

Table 3: The word-level perplexities for each language computed on tri-gram language model considering tokens (PPL-T) and stems (PPL-S) along with critical bottleneck dimensionality.

both correlation coefficients are high, only the correlation coefficient between the stem-based perplexity and the critical dimensionalities is statistically significant (this is not surprising as autoencoders were actually trained with stems rather than tokens). This result implies that there is actually a strong dependence between the perplexity of a language and its critical bottleneck dimensionality.

Mode	Correlation	p-value
tokens	0.95797	0.10339
stems	0.88834	**0.04168***

Table 4: The correlation between critical dimension for a language and its word-level perplexity. The p-value represents the two-tailed TTest values. * denotes the statistical significance (< 0.05).

6. Conclusions and Future Research Directions

In this work we have presented a comprehensive study on the use of autoencoders for modelling text data, in which differently from previous studies we focused our attention in the following issues:

- We explored the suitability of two different models bDA and rsDA for constructing deep autoencoder representations of text data at the sentence level.
- We proposed and evaluated two novel metrics which assess the reconstruction quality of an autoencoder with regards to the particular problem of text data representation.

- We proposed an automatic method to find the critical bottleneck dimensionality for text language representation, below which structural information is lost.

- We conducted a comparative evaluation across different languages and explored the relationship between the critical bottleneck dimensionality and language perplexity.

As a result of this study we have found that the bDA model is most suitable for constructing and training autencoders for handling text data at the sentence level. We also found that our defined SPI (Structure Preservation Index) metric allows for a better discrimination and identification of the critical bottleneck dimensionality, which happens to be different for different languages and exhibits a high and significant correlation coefficient with language perplexity.

As future work, we want to study the suitability of our proposed metrics, especially SPI, as error metric during the autoencoder fine tuning stage. If this metric can be used along with back-propagation, we envisage a new generation of text-oriented autoencoders able to provide a much better characterization of the linguistic phenomenon in text data.

Acknowledgments

The work of the first and third authors was carried out in the framework of the WIQ-EI IRSES project (Grant No. 269180) within the FP 7 Marie Curie, the DIANA APPLICATIONS Finding Hidden Knowledge in Texts: Applications (TIN2012-38603-C02-01) project and the VLC/CAMPUS Microcluster on Multimodal Interaction in Intelligent Systems.

References

Banchs, R. E., Kaltenbrunner, A., 2008. Exploiting mds projections for cross-language ir. In: Myaeng, S.-H., Oard, D. W., Sebastiani, F., Chua, T.-S., Leong, M.-K. (Eds.), SIGIR. ACM, pp. 863–864.

Blei, D. M., Ng, A. Y., Jordan, M. I., Mar. 2003. Latent dirichlet allocation. J. Mach. Learn. Res. 3, 993–1022.
URL http://dl.acm.org/citation.cfm?id=944919.944937

Bourlard, H., Kamp, Y., Sep. 1988. Auto-association by multilayer perceptrons and singular value decomposition. Biological Cybernetics 59 (4), 291–294.
URL http://dx.doi.org/10.1007/bf00332918

Brown, P. F., Pietra, V. J. D., Mercer, R. L., Pietra, S. A. D., Lai, J. C., Mar. 1992. An estimate of an upper bound for the entropy of english. Comput. Linguist. 18 (1), 31–40.
URL http://dl.acm.org/citation.cfm?id=146680.146685

Cox, T., Cox, M., 2001. Multidimensional Scaling. CRC/Chapman and Hall.

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., Harshman, R. A., 1990. Indexing by latent semantic analysis. JASIS 41 (6), 391–407.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., Bengio, S., Mar. 2010. Why does unsupervised pre-training help deep learning? J. Mach. Learn. Res. 11, 625–660.
URL http://dl.acm.org/citation.cfm?id=1756006.1756025

Gabrilovich, E., Markovitch, S., 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In: Proceedings of the 20th international joint conference on Artifical intelligence. IJCAI'07. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 1606–1611.
URL http://dl.acm.org/citation.cfm?id=1625275.1625535

Gupta, P., Bali, K., Banchs, R. E., Choudhury, M., Rosso, P., 2014. Query expansion for mixed-script information retrieval. In: Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval. SIGIR '14. ACM, New York, NY, USA, pp. 677–686.
URL http://doi.acm.org/10.1145/2600428.2609622

Hinton, G., Salakhutdinov, R., 2006. Reducing the dimensionality of data with neural networks. Science 313 (5786), 504 – 507.

Hinton, G. E., 2002. Training products of experts by minimizing contrastive divergence. Neural Computation 14 (8), 1771–1800.

Hofmann, T., 1999. Probabilistic latent semantic indexing. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. SIGIR '99. ACM, New York, NY, USA, pp. 50–57.
URL http://doi.acm.org/10.1145/312624.312649

Nie, J.-Y., Simard, M., Isabelle, P., Durand, R., 1999. Cross-language information retrieval based on parallel texts and automatic mining of parallel texts from the web. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. SIGIR '99. ACM, New York, NY, USA, pp. 74–81.
URL http://doi.acm.org/10.1145/312624.312656

Platt, J. C., Toutanova, K., tau Yih, W., 2010. Translingual document representations from discriminative projections. In: EMNLP. ACL, pp. 251–261.

Quartz, S. R., Sejnowski, T. J., 1997. The neural basis of cognitive development: A constructivist manifesto. Behavioral and Brain Sciences 20 (04).
URL http://dx.doi.org/10.1017/S0140525X97001581

Salakhutdinov, R., Hinton, G., Jul. 2009a. Semantic hashing. Int. J. Approx. Reasoning 50 (7), 969–978.
URL http://dx.doi.org/10.1016/j.ijar.2008.11.006

Salakhutdinov, R., Hinton, G. E., 2009b. Replicated softmax: an undirected topic model. In: Bengio, Y., Schuurmans, D., Lafferty, J. D., Williams, C. K. I., Culotta, A. (Eds.), NIPS. Curran Associates, Inc., pp. 1607–1614.

Socher, R., Pennington, J., Huang, E. H., Ng, A. Y., Manning, C. D., 2011. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP).

Utgoff, P. E., Stracuzzi, D. J., Oct. 2002. Many-layered learning. Neural Comput. 14 (10), 2497–2529.
URL http://dx.doi.org/10.1162/08997660260293319

Xu, J., Croft, W. B., 1996. Query expansion using local and global document analysis. In: Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval. SIGIR '96. ACM, New

York, NY, USA, pp. 4–11.
URL http://doi.acm.org/10.1145/243199.243202