Semantic Preserving Siamese Autoencoder for Binary Quantization of Word Embeddings

Wouter Mostard
University of Groningen
Groningen, Netherlands
w.mostard@rug.nl

Lambert Schomaker
University of Groningen
Groningen, Netherlands
l.r.b.schomaker@rug.nl

Marco Wiering
University of Groningen
Groningen, Netherlands
m.a.wiering@rug.nl

ABSTRACT

Word embeddings are used as building blocks for a wide range of natural language processing and information retrieval tasks. These embeddings are usually represented as continuous vectors, requiring significant memory capacity and computationally expensive similarity measures. In this study, we introduce a novel method for semantic hashing continuous vector representations into lower-dimensional Hamming space while explicitly preserving semantic information between words. This is achieved by introducing a Siamese autoencoder combined with a novel semantic preserving loss function. We show that our quantization model induces only a 4% loss of semantic information over continuous representations and outperforms the baseline models on several word similarity and sentence classification tasks. Finally, we show through cluster analysis that our method learns binary representations where individual bits hold interpretable semantic information. In conclusion, binary quantization of word embeddings significantly decreases time and space requirements while offering new possibilities through exploiting semantic information of individual bits in downstream information retrieval tasks.

CCS CONCEPTS

• **Information systems** → **Information retrieval**; • **Computing methodologies** → Neural networks; **Learning latent representations**.

KEYWORDS

Semantic hashing, Siamese autoencoder, Representation learning

ACM Reference Format:
Wouter Mostard, Lambert Schomaker, and Marco Wiering. 2021. Semantic Preserving Siamese Autoencoder for Binary Quantization of Word Embeddings. In *Proceedings of NLPIR 2021: 5th International Conference on Natural Language Processing and Information Retrieval (NLPIR).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

NLPIR, December 17–20, 2020, Online

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Dense vector representations of words, or word embeddings [17], have been applied to a wide range of downstream natural language processing and information retrieval tasks. These include creating semantic document representations [20] and performing query-document matching [9]. The vectors are learned from the co-occurrence of word representations in large text corpora. After training the word vectors hold syntactic and semantic information about words.

Using continuous representations in large-scale information retrieval tasks, such as web information retrieval, has two important disadvantages. First, the continuous representations have significant memory requirements since each element of the vector is represented as a floating-point number. As a result, fewer documents can be held in main memory. Second, comparing continuous vectors requires floating-point similarity measures, such as cosine similarity or dot-product. These are more expensive than for example calculating the Hamming distance. Both disadvantages significantly hinder the applicability of continuous representations in large-scale information retrieval tasks. A popular approach for decreasing the computational requirements while retaining semantic information is called *semantic hashing* [23].

Our research specifically focuses on semantic hashing of continuous word representations into lower-dimensional binary Hamming space. Representing word vectors in binary Hamming space offers two distinct advantages. First, each element of a binary representation can be represented by a single bit, significantly decreasing memory requirements. Second, binary vectors allow for comparison of vectors using simple bitwise operators instead of more expensive floating-point calculations. This allows for a significant increase in document comparison speed. Various methods for hashing word vectors into lower-dimensional binary Hamming space have been proposed. These include setting a hard threshold after performing random projection locality sensitivity hashing (LSH) [24] and applying the Heaviside step function to the latent layer of an autoencoder [28]. One significant disadvantage of the above-mentioned methods is that the preservation of semantic similarity between word representations in input space and latent space is not explicitly enforced. For the autoencoder, similarity is only implicitly imposed by including a reconstruction error such that the most salient information is retained in latent space. In order to enforce a topology-preserving transformation, additional constraints are required.

In this research, we seek to preserve input topology by introducing a novel hashing method that directly enforces retainment of semantic information through a semantics-preserving loss function.

In this paper, several contributions are made. First, we propose a Siamese autoencoder architecture [29] that directly enforces the

conservation of the semantic similarity between word representations in Euclidean input space and lower-dimensional Hamming latent space. Second, our model is compared to several state-of-the-art methods using various standard semantic similarity and sentence classification tasks. Last, applicability to information retrieval is assessed by performing top-k document retrieval and clustering using a mixture of Bernoulli distributions. Results show that our semantic preserving Siamese autoencoder displays competitive results for the semantic similarity datasets while exhibiting a significant improvement for the information retrieval tasks. In summary, our contributions are as follows:

(1) We propose a Siamese autoencoder model employing a semantic preserving loss function that directly enforces retainment of semantic similarity between pairs of word representations in Euclidean input space and Hamming latent space.
(2) Our proposed model is tested on ten semantic similarity and sentence classification tasks. Furthermore, the applicability to information retrieval is determined by top-K document retrieval on three text document datasets and clustering.
(3) Clustering of text documents is qualitatively evaluated showing that through additive vector quantization binary codes are learned where individual bits exhibit interpretable semantic information.

The rest of this paper is organized as follows. Section 2 presents related work in semantic hashing and autoencoder architectures. Then Section 3 explains the proposed method and how we evaluate it compared to baseline methods. Section 4 then presents the results of the performed experiments. The paper concludes with a discussion and conclusion in Section 5.

2 BACKGROUND

In this section we describe relevant background information in semantic hashing, autoencoders, and Siamese networks.

2.1 Semantic Hashing

Semantic hashing is a method for hashing documents in a way such that semantically similar documents are hashed close to each other in a lower-dimensional latent space. Hashing methods can broadly be grouped into two categories: *data-independent* and *data-dependent* methods.

A simple method for data independent hashing of word embeddings was introduced in [24]. A randomly initialized weight matrix $W^{N\times M} \sim \text{Uniform}(-\frac{1}{\sqrt{N}}, \frac{1}{\sqrt{N}})$ is used to linearly transform a N-dimensional word embedding $x_i \in \mathbb{R}^N$ into an M-dimensional latent representation $h_i = x_i \cdot W$. Binary codes are subsequently obtained by applying a hard element-wise threshold, i.e. $b_i = sign(h_i \geq 0)$.

Classical examples of data-dependent hashing methods are the Restricted Boltzmann Machine [23] and Latent Semantic Analysis [8] that are typically applied to a term-document matrix. Recently, [28] proposed a model for hashing word representations using an autoencoder architecture. [18] proposed using a semantic similarity measure. This is different than our hashing method considering they only take the relative distance between word pairs into account while we directly try to minimize this using a straight-through estimator instead of hard thresholding. Other approaches are for example [13] where adaptive compression of embeddings into discrete codes is performed using the Gumbel-softmax trick. [26] tried to perform binary quantization using a triangle similarity measure. In [33] a quantization method is introduced by adding a L1 loss on the large-scale latent representations such that the representations are sparse and suitable for inverted indexing. In this research we focus on retaining semantic information in individual bits.

2.2 Autoencoders

One commonly used approach for incorporating information from input vectors is using an autoencoder architecture. An autoencoder consists of an encoder E which transforms an N-dimensional input vector into an M-dimensional latent representation and a decoder D that transforms the latent representation back into the original input space. Learning is achieved by adding a reconstruction loss which is propagated back through the network. A frequently used reconstruction loss is the mean squared error which is defined as:

$$\mathcal{L}_{\text{rec}}(\vec{x}_i, \vec{r}_i) = \frac{1}{N}\sum_{k=1}^{N}(x_{i_k} - \hat{r}_{i_k})^2 \tag{1}$$

Where x_{i_k} and $\hat{r}_{i_k}$ represent the kth bit of the input and reconstructed vector respectively. For most practical applications an undercomplete autoencoder is used, meaning that M is smaller than N. This forces the autoencoder to represent the most salient details from the input data into latent space from which the decoder can reconstruct the original input vector.

Several methods have been proposed to perform semantic hashing using autoencoders. The authors in [28] proposed a method where both the encoder and the decoder consist of a single weight sharing matrix $W^{N\times M}$, where the decoder uses the transpose of W to reconstruct the vector. Binary codes of size M are obtained by $h(x \cdot W)$, where $h(\cdot)$ is an element-wise Heaviside step function. Considering the Heaviside step function is non-differentiable, learning is achieved by using the gradients of the decoder weights $(\frac{\partial(x_i-\hat{r}_i)^2}{\partial W})$ to update the encoder weights. One disadvantage of this approach is that it potentially leads to significant quantization error. In order to decrease quantization error and allow for end-to-end learning the Gumbel-softmax reparametrization trick is proposed by [11]. This reparametrization trick is applied by [25] to learn discrete latent variables. Another approach is based on variational inference [15]. One significant disadvantages of the above mentioned approaches is that the relational structure in the input space is only implicitly retained in latent space.

2.3 Siamese Networks

A popular method for utilizing information from distinct input pairs is through Siamese neural networks [4]. A Siamese neural network transforms input pairs or triplets, into a latent representation through a weight sharing sub-network. Using weight sharing sub-networks allows the model to learn distinct similarities or differences between input pairs by adopting a contrastive or triplet loss function. This method has been applied to various discrimination tasks, including logo detection [30] and face recognition [27].

Recently, [29] proposed a, Siamese autoencoder for dimensionality reduction while preserving the Mahalanobis distance between input pairs. This is achieved by adding a multidimensional scaling loss function which objective is to minimize the difference between Euclidean distances between pairs in input space and latent space. Some research has been conducted into using Siamese networks in the context of quantization. For example [19] applied a bi-directional LSTM with a Siamese architecture for job title normalization and [22] applied it to predict the relatedness of sentence pairs. Both methods primarily focus on predicting similarity among input pairs. To our knowledge Siamese networks have not yet been applied to quantization of word embeddings nor has its applicability to information retrieval tasks been assessed.

3 METHOD

In this section we describe our proposed model. Furthermore, we describe the evaluation procedure that is used to compare our model to the baseline methods.

3.1 Data

FastText[1] [16] word representations serve as input for the proposed quantization methods. No preprocessing is performed.

3.2 Similarity Preserving Autoencoder

In order to explicitly preserve the semantic similarity between word representations, a model was developed that tries to minimize the differences between the similarity of two word representations in input space and latent space. A fitting approach to solve this problem is using Siamese autoencoders. Our Siamese autoencoder consists of several parts. First, an encoder E transforms the input representations into a lower-dimensional latent representation using a standard neural network architecture. Second, a decoder D is required to transform the binary latent representations back into the original continuous space. Third, a loss function is used to directly enforce that the difference between similarity for the continuous input representations and binary latent representations is minimized.

Let $\vec{x}_i$ and $\vec{x}_j$ be two continuous N-dimensional vector representations. In [28], it was shown that competitive binary representations can be obtained by using a single linear transformation. Comparable with that research we used a single weight-sharing matrix $W^{N\times M}$ in each encoder to linearly transform both input pairs into lower-dimensional space:

$$\vec{h}_i = W \cdot \vec{x}_i + b$$

where b is a bias term and $N \times M$ are the number of input dimensions and target latent representation size respectively. Note that these latent representations are still continuous. Instead of using a non-differentiable Heaviside function, we utilize the differentiable Straight-through estimator [3] in order to obtain the binary representations $\vec{b}_i$.

The goal of our decoder is to transform the binary representations $\vec{b}_i$ back into the original continuous space. In [25], it was shown that adopting a coding scheme of additive quantization can be utilized for reconstructing the input vectors using K-dimensional categorical variables. In our case we applied additive vector quantization [2] of individual codewords such that our reconstructed vector is the sum of the code words assigned to the individual bits. Formally, let $\vec{b}_i$ be a binary vector and $A^{M\times N}$ be a codebook consisting of M codewords in N dimensions. The reconstructed vectors are then obtained by:

$$\hat{r}_i = \vec{b}_i \cdot A \tag{2}$$

This enforces our model to learn representative codewords since the reconstruction is a sum of only the codewords that are activated by the bits in the binary vector. Furthermore, note that the codebook is shared between both decoders.

Two loss functions are applied in order to train the proposed model. First, the reconstruction loss is calculated as the average of the mean squared error loss of both reconstructed vectors:

$$\mathcal{L}_{\text{recon}} = \frac{\mathcal{L}_{\text{rec}}(\vec{x}_i, \vec{r}_i) + \mathcal{L}_{\text{rec}}(\vec{x}_j, \vec{r}_j)}{2} \tag{3}$$

Note that this is comparable with a standard autoencoder architecture and no semantic information in the input space is used for training. In order to exploit this information, we use a similar method as proposed in [29]. However, instead of optimizing the retainment of the Mahalanobis distance, we are interested in the retainment of the semantic similarity between pairs of vectors. A frequently used measure for semantic similarity between continuous vectors x_i and x_j is given by the Cosine similarity:

$$\text{Cosine}(\vec{x}_i, \vec{x}_j) = \frac{\vec{x}_i \cdot \vec{x}_j}{||\vec{x}_i||||\vec{x}_j||}$$

Now let $\vec{b}_i$ and $\vec{b}_j$ be the binary latent representations of $\vec{x}_i$ and $\vec{x}_i$ respectively. A suitable measure for calculating the similarity of two binary vectors is given by the normalized Hamming similarity (NHS) which is given by:

$$NHS(\vec{b}_i, \vec{b}_j) = 1 - \sum_{k=0}^{M} \frac{|b_{i_k} - b_{j_k}|}{M} \tag{4}$$

Note however that the cosine similarity has a range of $[-1, 1]$ while the normalized Hamming similarity has a range of $[0, 1]$. In order to overcome this we propose to use the angular similarity which is given by:

$$\text{Angular}(\vec{x}_i, \vec{x}_j) = 1 - \frac{cos^{-1}(\frac{\vec{x}_i \cdot \vec{x}_j}{||\vec{x}_i||||\vec{x}_j||})}{\pi} \tag{5}$$

Which has the desirable range of $[0, 1]$. The goal is to preserve the semantic similarity between vectors after they have been transformed into lower-dimensional Hamming space. This can explicitly be enforced as follows:

$$\mathcal{L}_{\text{preserve}} = (NHS(\vec{b}_i, \vec{b}_j) - \text{Angular}(\vec{x}_i, \vec{x}_j))^2 \tag{6}$$

Where NHS and Angular are given by equations 4 and 5 respectively. The maximum loss is obtained when the semantic similarity scores of two representations are opposite while it is zero when both semantic similarity measures agree.

[1] https://fasttext.cc/docs/en/english-vectors.html

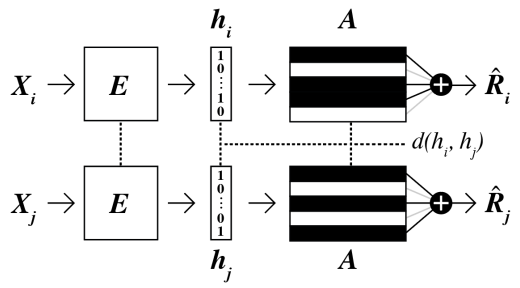

Figure 1: Proposed Siamese autoencoder architecture. The dashed lines between E and A indicate weight sharing.

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{preserve}} \tag{7}$$

See Figure 1 for a graphical depiction of the proposed model.

3.3 Evaluation

Our model is compared with a strong data-independent and data-dependent baseline method. For the data independent method we use the random projection Locality Sensitivity Hashing (LSH) method proposed in [24]. The autoencoder architecture proposed in [28] (NLL) is used as a data-dependent baseline. In order to assess the influence of our semantic preserving loss, we compare our model with (AE +SP) and without (AE) the semantic preserving loss function. Binary codes of size 64, 128, and 256 are chosen considering we are explicitly interested in hashing into lower-dimensional spaces which are in adequacy with CPU register sizes.

Evaluation of the semantic retainment of word-level binary representations is performed using various semantic and syntactic similarity tasks. The following datasets were used: the MEN Test Collection [5], Card-660 Rare Words [21], SimLex-999 [10], and WordSim353 [1]. SentEval[2] [6] with default settings is used for downstream sentence-level evaluation tasks. The tasks included are: product reviews (CR), opinion polarity detection (MPQA), sentiment analysis (MR, STS14), and subjectivity classification (SUBJ). Sentence embeddings are obtained by taking a sum of the word representations. No normalization is performed prior to hashing

Cosine similarity is used to compute the semantic similarity for continuous representations and the normalized Hamming similarity for the binary representations. Correlation with the human-annotated similarity is evaluated using Pearson r.

Precision at K (P@K) is used to measure retrieval performance. P@K is defined as the number of relevant documents divided by K. All used datasets contain class labels thus a document is deemed relevant if it holds the same class label as the test document. In this experiment, we retrieve the top 100 most similar documents, i.e. $K = 100$, for each test document. Three publicly available text document collections are used to assess document retrieval. First, Reuters215782 is a collection of 10,788 news articles distributed over 90 categories. Only the categories that have more than 100 documents in the training set are considered. 20NG is a text corpus consisting of 18,828 newsgroup articles almost uniformly distributed over 20 newsgroups. AG news[3] is a large-scale collection of news documents. The categories: world, sports, business, and science have been selected for evaluation. 17,000 documents have been uniformly sampled for the dataset, 15 thousand for training and two thousand for testing. For all data sets, the documents have been trimmed to the first thousand characters and stop words are removed. No other pre-processing is performed. Document representations are obtained by taking the sum of the word vectors. During evaluation, each test document is used as a query document

3.4 Training Procedure

The LSH method is initialized as proposed in [24]. No further training is required. Binary codes are obtained by thresholding the latent representation at 0. The NLL and our Siamese autoencoder model are optimized using the Adam[14] optimizer with a learning rate of 10^{-4} and batch size of 128. The regularization parameter λ for the NLL loss function has been set to 1. Training is halted when no significant change in the reconstruction loss is observed over a period of 10 epochs with a maximum number of epochs set at 150. For the Siamese autoencoder positive samples are uniformly sampled from the 10 nearest neighbors of the anchor word representation. Sampling is performed by choosing a positive example with 20% probability or a random word vector otherwise.

4 RESULTS

In this section the results are presented. First, our model is quantitatively compared to several baseline methods using word similarity, classification, and document retrieval datasets. Furthermore, the decorrelating effect of additive vector quantization and the average correlation for different bit sizes is compared. Lastly, we qualitatively assess the interpretability of the binary codes.

4.1 Semantic Similarity and Classification

Table 1 shows the results for the different baseline methods and our proposed Siamese autoencoder with and without the semantic preserving loss function. First, we discuss the word semantic similarity results which are shown in the top half of the table. Several observations can be made from these results. First, except for the Rare Words dataset, the 256-bit AE or AE + SP model achieves the highest correlation score with the human-annotated word similarity score. The best performing 256-bit representations show a mean degradation of 7.7% over the continuous representations. It should be noted that the representations are 9.4 times smaller in terms of memory usage. Second, the data-independent LSH method shows competitive results in most word similarity datasets.

The results for several standard sentence classification (CR, MR, MPQA, SUBJ) and one semantic textual similarity (STS14) datasets are shown in the bottom half of Table 1. The AE or AE + SP models consistently yield better results for all datasets and across all bit sizes. This is especially predominant for the 64 and 128-bit representations. The best performing 256-bit representations show a mean degradation of 4.9% over the continuous representations. Interestingly, the 256-bit representations of the semantic preserving

[2]https://github.com/facebookresearch/SentEval

[3]http://groups.di.unipi.it/ gulli/AG_corpus_of_news_articles.html

Table 1: Word similarity and classification results of the various proposed methods. Scores of original 300-dimensional floating-point representations are in placed between parentheses. In the top half the Spearman's rank correlation for word similarity and word analogy are shown. In the bottom half results of downstream classification datasets are shown. For CR, MPQA, MR, and SUBJ the accuracy is shown and for STS14 the Spearman's rank correlation. Cosine similarity is used for comparing continuous representations and Hamming similarity for binary representations

	MEN (0.82)			RW (0.57)			SimLex (0.52)			SimVerb (0.44)			WS353 (0.74)		
	64	128	256	64	128	256	64	128	256	64	128	256	64	128	256
LSH	0.56	0.67	0.73	**0.40**	**0.47**	0.48	**0.36**	0.38	0.47	**0.30**	**0.30**	0.37	0.53	0.55	0.67
NLL	0.57	0.68	0.75	0.39	0.46	**0.52**	0.35	0.37	0.46	0.26	**0.30**	**0.39**	0.49	0.61	0.64
AE	**0.64**	0.71	0.76	**0.40**	0.46	0.50	0.31	0.38	0.45	0.24	**0.30**	0.37	0.53	0.64	**0.70**
AE + SP	0.62	**0.74**	**0.77**	0.37	0.45	0.51	0.27	**0.39**	**0.48**	0.24	**0.30**	**0.39**	**0.58**	**0.68**	**0.70**

	CR. (80.6)			MPQA (88.0)			MR (78.2)			STS14 (0.63)			SUBJ (92.3)		
	64	128	256	64	128	256	64	128	256	64	128	256	64	128	256
LSH	67.5	68.2	73.3	77.2	81.9	84.5	63.5	67.7	71.0	0.49	0.54	0.58	76.5	82.3	86.9
NLL	65.5	70.8	74.8	77.8	81.5	84.3	60.7	67.9	71.4	0.39	0.54	0.60	74.4	84.2	85.9
AE	**72.4**	**74.0**	75.3	**80.9**	**83.8**	85.3	**68.7**	71.5	**74.0**	**0.53**	**0.59**	0.63	84.1	**87.4**	89.0
AE + SP	71.7	73.2	**75.7**	80.7	83.3	**85.4**	68.6	**71.8**	**74.0**	0.52	**0.59**	**0.64**	**84.7**	86.5	**89.8**

autoencoder model perform even slightly better than the continuous representations on the textual similarity dataset. This may be attributed to the fact that the cosine similarity compares documents on a unit sphere while hashing is performed on unnormalized representations.

4.2 Document Retrieval

Given that the binary representations show competitive results on sentences we are interested in how well it would perform on larger text documents and on traditional information retrieval tasks.

Table 2 shows the P@K scores for three text document collections. The scores for the continuous representations are shown in the *Cont.* column. We make several interesting observations. First, the autoencoder with the semantic preservation loss performed best in all but two retrieval tasks at various bit sizes. Furthermore, our model performed statistically significant better for $p \leq 0.05$ compared to the best performing baseline method at each individual task. Second, for the best performing 256-bit semantic preserving autoencoder the degradation with respect to the continuous representations for the 20 newsgroup and Reuters dataset is 5% and 2.7% respectively. Interestingly, the 256-bit semantic preserving model has a 4.2% higher P@K score over the continuous representations for the AG news dataset. A similar phenomenon has been observed for the STS14 dataset in Table 1. Third, the semantic preserving model performs better than the vanilla Siamese autoencoder in all but two retrieval tasks and is statistically significant for $p \leq 0.05$ for three of the nine tasks.

4.2.1 Additive Quantization. One finding from Table 2 is the *increased* performance of the 256-bit representations over the continuous baseline on the AG news dataset. Possibly this is due to the decorrelating effect that our additive vector quantization decoder has. By not applying a non-linear activation at the output layer as applied in [28] and disabling the bias we enforce the reconstruction of the original vector to be constructed from N activated codewords.

Table 2: Precision at 100 on three standard text collections. The scores using the original 300-dimensional continuous word representations are shown in the *cont* column. $\star$ is used to depict scores that are statistically significant using the binomial test where $p \leq 0.05$ compared with the best performing baseline model. Models where the semantic preservation score is significantly better compared to the vanilla autoencoder are marked with an underscore.

	cont.	LSH	NLL	AE	AE + SP
20NG	0.40				
64	-	0.15	0.12	0.27*	**0.29***
128	-	0.21	0.23	0.33*	**0.36***
256	-	0.26	0.28	**0.38***	**0.38***
AG news	0.70				
64	-	0.44	0.39	0.67*	**0.68***
128	-	0.52	0.51	0.71*	**0.71***
256	-	0.59	0.60	0.72*	**0.73***
Reuters	0.74				
64	-	0.59	0.52	0.62*	**0.64***
128	-	0.59	0.63	**0.68***	**0.68***
256	-	0.65	0.67	0.71*	**0.72***

In order to assess whether this was due to vector quantization, we try to reconstruct the original document vector not with all the available bits but only with N% of the most activated bits in a given test set. Figure 2 shows the results of the P@K scores evaluated at different N for the best performing 256-bit models.

Figure 2 shows that the codebook models, i.e. AE+SP and AE, are already able to perform on par with the continuous baseline with only approximately 50% of the bits enabled. The result flattens out after 50%, indicating that the least activated dimensions add limited semantic information. This ability to reconstruct semantic document vectors from a limited number of codewords could be

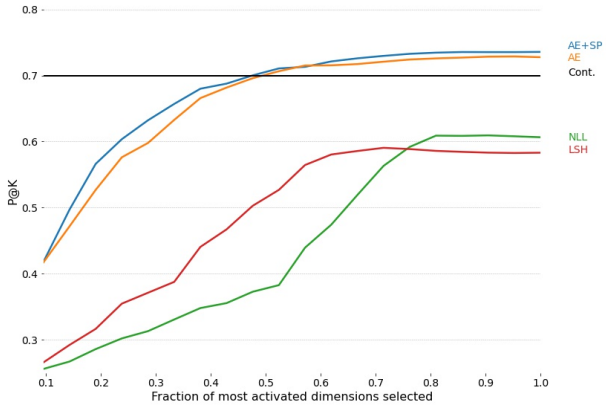

Figure 2: Influence of P@K performance when selecting top N most activated bits for the various methods on the NG news dataset. Cont. depicts the P@K score for the original continuous vectors

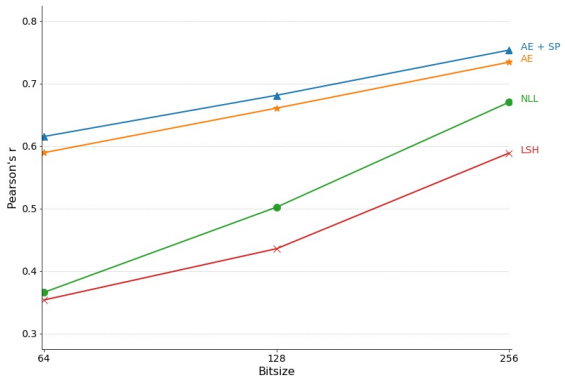

Figure 3: Mean correlation of 100 continuous representations with the binary representations for different bit sizes

indicative that our hashing methods perform automatic regularization such that only a few codewords are representative of the target classes. This effect is qualitatively evaluated in Section 4.4.

4.3 Average Correlation

One of the main advantages of the proposed methods is that latent representations of arbitrary size can be constructed. We were interested in determining how well the latent representations of different sizes are able to retain word-level correlation with the original continuous representations. In order to evaluate this, we extracted 100 test words and 20,000 randomly sampled evaluation words. For each test word the mean Pearson's r was calculated with the continuous representations over all evaluation words for different bit sizes. The result is depicted in Figure 3. Our Siamese autoencoder models start with a mean correlation of approximately 0.60 for the 64-bit representation and monotonically increases to approximately 0.74 for the 256-bit representation. The LSH and NLL method start with an average correlation of approximately 0.35 at the 64-bit representation. The baseline methods show a mean correlation of approximately 0.65 and 0.59 for the NLL and LSH method respectively at 256-bits. This shows that the difference between mean correlation for the baseline methods and our approach is most present at the lower-dimensional representations.

4.4 Qualitative Analysis

In order to qualitatively assess the generated binary representations, two experiments were performed. First, we applied clustering to determine the natural clustering of the binary representations. Second, individual bits from the cluster prototypes were utilized in order to illustrate the interpretability of individually learned code words.

4.4.1 Clustering. First, we were interested in clustering text documents in lower-dimensional Hamming space. We used the AG newsgroup which consists of news articles on four different topics: Science, World, Sports, and Business. The 64-bit binary representations were generated using the best performing semantic preserving Siamese autoencoder. In order to cluster the binarized representations, we apply a mixture of Bernoulli distributions [12].

Table 3: Titles of the nearest neighbors from cluster prototypes shown in Figure 4

DocID	Cluster	Title
1605	Science	Business Objects to bundle IBM tools
1288	World	American says US backed his jail
487	Sports	Huey Exits Early
1824	Business	Consumer Drop, Industry Output Up

The number of Bernoulli distributions was set to 4 considering this is the number of topics in the dataset. Each cluster $\vec{\mu}_i$ was uniformly initialized $\vec{\mu}_i \sim \text{Uniform}(0.25, 0.75)$. Training halted when the negative log-likelihood did not increase with more than $\Delta 10e^{-4}$ over the entire dataset. After training, each cluster prototype $\vec{c}_i$ was obtained by setting a hard element-wise threshold at 0.50 on $\vec{\mu}_i$. See Figure 4 for a t-SNE plot of the first 2000 documents from the dataset. The cluster prototypes are marked with a star.

Table 3 shows the docID, class, and title of the nearest neighbor of each cluster prototype. Document 1605 seems to relate to business IT and document 1288 discusses world news. Documents 487 and 1824 seem to be about a sportsman quitting and an opinion piece about consumer prices respectively.

4.4.2 Bit activation. Finally, we were interested in determining whether our Siamese autoencoder learned semantic codewords in the codebook decoder. To determine this we retrieved the most activated bit from the 10 nearest neighbors from each cluster prototype. Since additive vector quantization is used to decode the binary representation the index of the most activated bit corresponds to the most activated code word in the decoder. Table 4 shows the 10

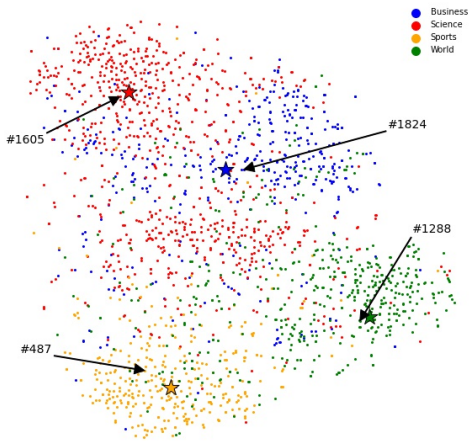

Figure 4: T-SNE plot of 2000 64-bit Hamming space representations sampled from the AG news dataset. Large star data points depict cluster prototypes obtained using Mixture of Bernoulli distributions

semantically most similar words given the highest activated bit for each cluster prototype.

Some interesting information can be deducted from these code words. First, we see that the fourth bit is frequently activated for the Science cluster which seems to be about IT technology. The 20th bit seems to detect verbs and adjectives that would frequently be used in political discussions. The first bit seems to be detecting names, which would often occur in a sports article. Bit number six seems to be about the unionization of the workforce. Some irrelevant terms, such as paleo and nachos, are also nearest neighbors of the sixth bit. The nearest neighbors of the codewords associated with the most activated bits seem relevant for the nearest neighbor of the cluster prototypes given in Table 3. For example, document 1824 seems to be about labor and industry, which is strongly activated with the bit as shown by words such as *unionism* and *Unionists*.

5 DISCUSSION AND CONCLUSION

Representing text documents into lower-dimensional Hamming space has gained increased interest due to the rise of large-scale information retrieval problems concerning documents and pages on the web. Representing documents in Hamming space allows for fast document comparison using bitwise operators and more efficient storage by representing each index as a single bit. A popular method for semantic hashing is the autoencoder architecture where the model transforms the input into an informative latent space such that the original vector can be reconstructed. One significant drawback of this approach is that the topology between input pairs is only implicitly retained through the reconstruction loss.

Table 4: Nearest neighbors for the codewords associated with the most activated bits in four clusters on the 64-bit binary representations on the AG news dataset. Index of the decoded bit is shown by #

Science (#4)	World (#20)	Sports (#1)	Business (#6)
business-oriented	staged	Drayden	unionism
open-source	triggered	Powhatan	predation
half-year	occurred	Safavieh	Unionists
multimedia	initiating	prAna	paleo
full-featured	right-wing	Vonn	nachos
multi-platform	traumatic	McDavid	unionization
fully-fledged	decisive	JustFab	scapegoating
enterprise-level	Soviet	Fabletics	Fluoridation
web-based	intensified	Cos.	Unionist
enterprise	subsequent	Whitner	unionist

In this paper, we demonstrate a method to overcome this problem by introducing a loss function that seeks to minimize the difference of semantic similarity of two word representations in Euclidean and in Hamming space. This is achieved by using a Siamese autoencoder architecture. The encoder consists of a single linear transformation while the decoder is based on additive vector quantization where the reconstruction is obtained by a linear combination of activated codewords. The added value of our approach is twofold: 1) rich semantic information is better retained in latent space, and 2) a single bit in a word vector represents valuable information that can be exploited in several tasks such as clustering and document retrieval.

Several interesting conclusions can be drawn from the experiments we conducted. First, the conservation of individual word semantics is assessed using several standardized data sets. Overall, our semantics-preserving Siamese autoencoder is the best performing model, with degradation of semantic information which is limited to 7% compared to continuous representations. For the sentence classification data sets the performance loss is on average 5%, showing that semantic hashing of bag-of-words representations of sentences can yield competitive results for various downstream natural language processing tasks. Last, we empirically show that our siamese autoencoder model learns semantically meaningful codewords that allow for competitive performance in document retrieval using only a small portion of the learned codewords.

We also assess the applicability of our hashing method on text document data sets. Using a mixture of Bernoulli distributions we qualitatively and quantitatively show that text documents that are semantically similar, are hashed close to each other in latent space. Our model outperforms the two baseline methods on all datasets and all bit sizes, in a statistically significant manner. The vanilla Siamese autoencoder is statistically significant outperformed in three of the nine tasks. In one task our proposed model even outperforms the continuous representations. Investigation shows that this effect can be attributed to having few activated bits that combined are representative of the document retrieval task. Qualitative inspection of individual bits shows that our model learns rich semantic bits which serve as a strong signal in retrieval. This is a

significant advantage over continuous vectors where individual floating-point numbers hold no semantics. This additional information could potentially be used to improve clustering by exploiting bitwise information. Some outliers are reported in nearest neighbors of highly activated bits. Possibly this is due to overfitting of the model and should be further evaluated.

Our work provides an initial step towards learning semantic and interpretable binary codes that are applicable for large-scale information retrieval systems. By using a Siamese autoencoder architecture it is possible to incorporate semantic information from input pairs to improve the semantic hashing of text documents into binary representations of any desired size. In general, our method allows for binary quantization with only an insignificant decrease of semantic information over the continuous vectors.

Future research should be conducted into the selection of informative input representations as is described in [31]. Another interesting research topic would be hashing word representations from different languages into a single space [7]. Finally, another interesting research direction would be to apply the Siamese autoencoder network to learn semantic hashes in a multimodal situation such as text-to-image retrieval [32].

REFERENCES

[1] Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and WordNet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL '09)*. Association for Computational Linguistics, USA, 19–27.

[2] Artem Babenko and Victor Lempitsky. 2014. Additive Quantization for Extreme Vector Compression. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, Columbus, OH, USA, 931–938. https://doi.org/10.1109/CVPR.2014.124

[3] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. *Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation*. Vol. abs/1308.3432. CoRR, arXiv. http://arxiv.org/abs/1308.3432 arXiv: 1308.3432.

[4] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a "Siamese" time delay neural network. In *Proceedings of the 6th International Conference on Neural Information Processing Systems (NIPS'93)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 737–744.

[5] Elia Bruni, Nam Khanh Tran, and Marco Baroni. 2014. Multimodal distributional semantics. *Journal of Artificial Intelligence Research* 49, 1 (Jan. 2014), 1–47.

[6] Alexis Conneau and Douwe Kiela. 2018. {S}ent{E}val: An Evaluation Toolkit for Universal Sentence Representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation ({LREC} 2018)*. European Language Resources Association (ELRA), Miyazaki, Japan, 1699–1704. https://www.aclweb.org/anthology/L18-1269

[7] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word Translation Without Parallel Data. https://openreview.net/forum?id=H196sainb

[8] Susan T. Dumais. 2004. Latent semantic analysis. *Annual Review of Information Science and Technology* 38, 1 (2004), 188–230. https://doi.org/10.1002/aris.1440380105

[9] Lukas Galke, Ahmed Saleh, and Ansgar Scherp. 2017. *Word Embeddings for Practical Information Retrieval*. Gesellschaft für Informatik, Bonn. https://doi.org/10.18420/in2017_215 Accepted: 2017-08-28T23:47:39Z ISSN: 1617-5468.

[10] Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with genuine similarity estimation. *Computational Linguistics* 41, 4 (Dec. 2015), 665–695. https://doi.org/10.1162/COLI_a_00237

[11] Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical Reparameterization with Gumbel-Softmax. https://openreview.net/forum?id=rkE3y85ee

[12] Ankur Kamthe, Miguel Á. Carreira-Perpinñán, and Alberto E. Cerpa. 2011. Adaptation of a mixture of multivariate Bernoulli distributions. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Volume Two (IJCAI'11)*. AAAI Press, Barcelona, Catalonia, Spain, 1336–1341.

[13] Yeachan Kim, Kang-Min Kim, and SangKeun Lee. 2020. Adaptive Compression of Word Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 3950–3959. https://doi.org/10.18653/v1/2020.acl-main.364

[14] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. http://arxiv.org/abs/1412.6980

[15] Francisco Mena and Ricardo Ñanculef. 2019. A Binary Variational Autoencoder for Hashing. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications (Lecture Notes in Computer Science)*, Ingela Nyström, Yanio Hernández Heredia, and Vladimir Milián Núñez (Eds.). Springer International Publishing, Cham, 131–141. https://doi.org/10.1007/978-3-030-33904-3_12

[16] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in Pre-Training Distributed Word Representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA), Miyazaki, Japan, 52–55. https://www.aclweb.org/anthology/L18-1008

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13)*. Curran Associates Inc., Red Hook, NY, USA, 3111–3119. event-place: Lake Tahoe, Nevada.

[18] Samarth Navali, Praneet Sherki, Ramesh Inturi, and Vanraj Vala. 2020. Word Embedding Binarization with Semantic Information Preservation. In *Proceedings of the 28th International Conference on Computational Linguistics*. International Committee on Computational Linguistics, Barcelona, Spain (Online), 1256–1265. https://doi.org/10.18653/v1/2020.coling-main.108

[19] Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. 2016. Learning Text Similarity with Siamese Recurrent Networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*. Association for Computational Linguistics, Berlin, Germany, 148–157. https://doi.org/10.18653/v1/W16-1617

[20] Giannis Nikolentzos, Polykarpos Meladianos, François Rousseau, Yannis Stavrakas, and Michalis Vazirgiannis. 2017. Multivariate Gaussian Document Representation from Word Embeddings for Text Categorization. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Association for Computational Linguistics, Valencia, Spain, 450–455. https://www.aclweb.org/anthology/E17-2072

[21] Mohammad Taher Pilehvar, Dimitri Kartsaklis, Victor Prokhorov, and Nigel Collier. 2018. Card-660: Cambridge Rare Word Dataset - a Reliable Benchmark for Infrequent Word Representation Models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 1391–1401. https://doi.org/10.18653/v1/D18-1169

[22] Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2019. Semantic Textual Similarity with Siamese Neural Networks. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*. INCOMA Ltd., Varna, Bulgaria, 1004–1011. https://doi.org/10.26615/978-954-452-056-4_116

[23] Ruslan Salakhutdinov and Geoffrey Hinton. 2009. Semantic hashing. *International Journal of Approximate Reasoning* 50, 7 (July 2009), 969–978.

[24] Dinghan Shen, Pengyu Cheng, Dhanasekar Sundararaman, Xinyuan Zhang, Qian Yang, Meng Tang, Asli Celikyilmaz, and Lawrence Carin. 2019. Learning Compressed Sentence Representations for On-Device Text Processing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 107–116. https://doi.org/10.18653/v1/P19-1011

[25] Raphael Shu and Hideki Nakayama. 2018. Compressing Word Embeddings via Deep Compositional Code Learning. https://openreview.net/forum?id=BJRZzFlRb

[26] Haohao Song, Dongsheng Zou, Lei Hu, and Jieying Yuan. 2020. Embedding Compression with Right Triangle Similarity Transformations. In *Artificial Neural Networks and Machine Learning – ICANN 2020 (Lecture Notes in Computer Science)*, Igor Farkaš, Paolo Masulli, and Stefan Wermter (Eds.). Springer International Publishing, Cham, 773–785. https://doi.org/10.1007/978-3-030-61616-8_62

[27] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. 2014. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*. IEEE Computer Society, USA, 1701–1708. https://doi.org/10.1109/CVPR.2014.220

[28] Julien Tissier, Christophe Gravier, and Amaury Habrard. 2019. Near-Lossless Binarization of Word Embeddings. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, Honolulu, Hawaii, USA, January 27 - February 1, 2019*., Vol. 33. AAAI Press, Palo Alto, California USA, 7104–7111.

[29] Lev V. Utkin, Vladimir S. Zaborovsky, Alexey A. Lukashin, Sergey G. Popov, and Anna V. Podolskaja. 2017. A Siamese Autoencoder Preserving Distances for Anomaly Detection in Multi-robot Systems. In *2017 International Conference on Control, Artificial Intelligence, Robotics Optimization (ICCAIRO)*. IEEE Computer Society, Los Alamitos, CA, USA, 39–44. https://doi.org/10.1109/ICCAIRO.2017.17

[30] Camilo Vargas, Qianni Zhang, and Ebroul Izquierdo. 2020. One Shot Logo Recognition Based on Siamese Neural Networks. In *Proceedings of the 2020 International Conference on Multimedia Retrieval (ICMR '20)*. Association for Computing Machinery, New York, NY, USA, 321–325. https://doi.org/10.1145/3372278.3390734

[31] Chao-Yuan Wu, R. Manmatha, Alexander J. Smola, and Philipp Krähenbühl. 2017. Sampling Matters in Deep Embedding Learning. In *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Los Alamitos, CA, USA, 2859–2867. https://doi.org/10.1109/ICCV.2017.309 ISSN: 2380-7504.

[32] Huei-Fang Yang, Kevin Lin, and Chu-Song Chen. 2018. Supervised Learning of Semantics-Preserving Hash via Deep Convolutional Neural Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 2 (Feb. 2018), 437–451. https://doi.org/10.1109/TPAMI.2017.2666812 arXiv: 1507.00101.

[33] Hamed Zamani, Mostafa Dehghani, W. Bruce Croft, Erik Learned-Miller, and Jaap Kamps. 2018. From Neural Re-Ranking to Neural Ranking: Learning a Sparse Representation for Inverted Indexing. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 497–506. https://doi.org/10.1145/3269206.3271800