Semantic Text Encoding for Text Classification using Convolutional Neural Networks

Ignazio Gallo, Shah Nawaz, Alessandro Calefati
University of Insubria
Varese, Italy
Email: ignazio.gallo@uninsubria.it

***Abstract*—In this paper, we encode semantics of a text document in an image to take advantage of the same Convolutional Neural Networks (CNNs) that have been successfully employed to image classification. We use Word2Vec, which is an estimation of word representation in a vector space that can maintain the semantic and syntactic relationships among words. Word2Vec vectors are transformed into graphical words representing sequence of words in the text document. The encoded images are classified by using the AlexNet architecture. We introduced a new dataset named *Text-Ferramenta* gathered from an Italian price comparison website and we evaluated the encoding scheme through this dataset along with two publicly available datasets i.e. 20news-bydate and StackOverflow. Our scheme outperforms the text classification approach based on Doc2Vec and Support Vector Machine (SVM) when all the words of a text document can be completely encoded in an image. We believe that the results on these datasets are an interesting starting point for many Natural Language Processing works based on CNNs, such as a multimodal approach that could use a single CNN to classify both image and text information.**

***Index Terms*—text encoding; word2vec; deep convolutional neural network; text documents classification;**

I. Introduction

Semantics plays a crucial role to understand the meaning of a text document; humans can understand the semantics easily; however, for a computer semantics understanding is still a distant goal. State-of-the-art word embedding models such as Word2Vec [1] are a step forward to understand the semantics of the text. ok Word2Vec takes words or phrases from the vocabulary and map to vectors of real numbers. We can then perform mathematical operations on these vectors, leading to different applications in Natural Language Processing. Word2Vec objective function causes words that occur in similar contexts to have similar word embeddings. We exploit this observation in our work as shown in Fig. 1, where similar word embeddings can be transformed into visual domain with similar encoding which then can be used in text classification with CNNs.

Text classification is a common task in Natural Language Processing extensively studied, for example [2] and [3] tackled this problem using traditional classifiers such as Naive Bayes, K-Nearest Neighbor and SVM. Although these are not state-of-the-art methods, they are still used for text classification.

Recently, Convolutional Neural Networks achieved state-of-the-art results on image classification [4]. In this work we propose a new encoding scheme called *Semantic Text Encoding* or *ste2img* which encodes Word2Vec features of a text document in an image, by capitalizing the abilities of CNNs for classification. Our proposed encoding scheme captures typical computer vision location invariant property along with compositionality. In fact, a CNN intuitively can understand words from pixels, sentences from words, and more complex concepts from sentences. The results on three different datasets show that our scheme achieves high accuracy and outperforms text classification based on Doc2Vec and SVM in many configurations. We believe that our encoding scheme will provide a basis for a future text/image multimodal approach and text classification.

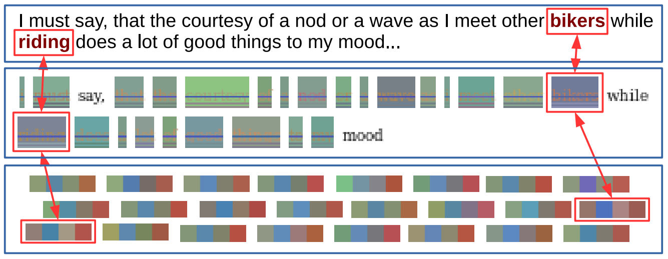

I must say, that the courtesy of a nod or a wave as I meet other bikers while riding does a lot of good things to my mood...
say,
while
mood

Fig. 1. When analyzing the text shown at the top, a human can understand that the two words ”bikers” and ”riding” are semantically related, and that the document talks about motorcycles. By combining the ability of Word2Vec to find semantic relationships between words and the ability of CNN in images classification, a computer can almost imitate those human capabilities. The visual words encoded by Word2Vec and displayed at the center and on the bottom of this figure, can be easily understood by a CNN. In this encoding, a 12 dimensions Word2Vec feature vector has been transformed into a sequence of 4 RGB colors.

II. Related work

In this section we discuss related works available in the literature. Zhang *et al.* [5], created a CNN model working with texts at the level of characters. They showed that their approach achieved low error rates compared to traditional approaches. However, in our work, instead of using a convolutional model developed specifically to deal with text classification, we designed a preprocess step which allows us to use the CNNs typically used for image classification. Kalchbrenner *et al.* [6] created a new model of CNN that used a specialized operation named Dynamic k-Max Pooling to classify sentences of varying length. Moreover, they provided a way to keep word order information intact unlike Bag-of-Words. In addition, they concluded their approach suffered

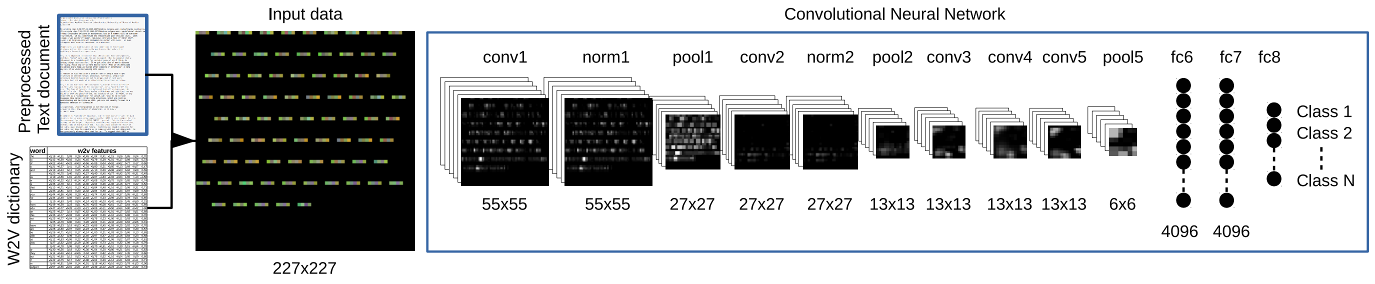


Fig. 2. Image of the proposed pipeline. Starting from a text, we encode each word with values extracted from a dictionary of feature vectors (W2V dictionary). Values obtained are interpreted as RGB values and represented in the final image. After this preprocess step, the image is classified by a properly trained CNN implementing the AlexNet architecture.

from complexity issues with larger dictionaries, resulting in the reduction of either the dimension of the vocabulary or the length of feature vectors. This leads us to consider that we should have the same issue; in fact, some of the datasets that we used are characterized by a large number of words, which forced us to limit the number of words especially for the 20news dataset.
Tang *et al.* [7] faced up the sentiment classification problem on Twitter datasets using an ad-hoc model made up of three neural networks to encode sentiment information from text into loss function. To achieve this goal, this study exploits the capabilities of Word2Vec model. They concluded that Word2Vec is not suitable for sentiment analysis tasks as it cannot differentiate the meaning between adjectives before a noun, for example "good" and "bad". On the other hand, we are interested in the relationships between words rather than polarity, and so this Word2Vec embedding is perfect for our purpose.

III. Proposed encoding scheme

The encoding method described in this section is simply used to analyze the behavior of the CNN, while varying the encoding parameters. We exploit Word2Vec [1] word embedding, that helps to reconstruct the semantics associated with a text document in an image and then the encoded image can be used in the classification process. As shown in Fig. 2, we used a dictionary $F(t_k, v_k)$ where each word t_k, used to train Word2Vec, is associated with a vector $v_k(t_k)$ of features obtained from a trained version of Word2Vec.

As shown in Fig. 2, to create the representation of a text document D_i, we start from a preprocessed version of D_i, by applying some transformations to the text. In this paper we tried to apply different preprocessing methods, and compared them with the raw text (see the graphical comparison in Fig. 5).

To encode all the words $t_k \in D_i$ into an image of size $W \times H$ we introduced a basic scheme that, after changing some parameters, allows us to obtain different arrangements of visual words $\hat{t}_k$ in the image. We introduce the concept of *super-pixel* as a square area of size $P \times P$ pixels with uniform color representing a contiguous sequence of features $(v_{k,j}, v_{k,j+1}, v_{k,j+2})$ extracted as a sub-vector of v_k. Each $v_{k,j}$ component is normalized with respect to all k, in such a way that it can assume values in the interval $[0 \ldots 255]$. Given a word t_k, a *visual word* $\hat{t}_k$ is a square area of $V \times V$ pixels that can contain a maximum number of super-pixels equal to $(V/P)^2$. The position $(0, 0)$ of each $\hat{t}_k$ was fixed in the upper left corner. Consequently, each $\hat{t}_k$ is placed in the following image coordinate (x, y):

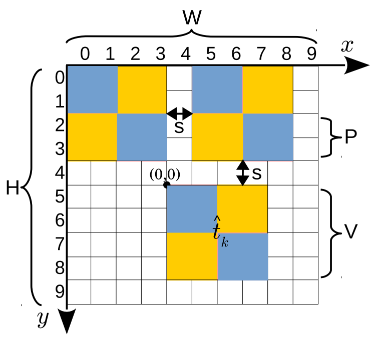
$$\begin{aligned} x &= k(V+s) \mod (W-V) \\ y &= (V+s)\tfrac{k(V+s)}{(W-V)} \end{aligned} \tag{1}$$

where s is the horizontal or vertical space in pixels existing between two close visual words. Fig. 3 shows an example of an image encoding a text document (on the left), and one example of encoded visual word (on the right). The visual words positioning described in equation (1), produces a wanted vertical misalignment, avoiding regularity in the encoding image. Some real examples of encoded images can be seen in Fig. 8.

With this encoding scheme, we can exploit CNN models typically used for image classification. As shown in Fig. 2, each convolutional layer produces different convolutive maps and each of them, from the closest layer to the input up to the last one, produces activation areas that show how the model is working to understand the semantics behind the text. The *conv1* layer in Fig. 2 shows activation areas near the individual superpixels, and as also showed by the convolutional kernels in Figure 4, this behavior does not depend on the size of kernels used for the input image. We believe that the first convolutional layer recognizes some particular features of visual words. Remaining CNN layers instead, aggregate these simple activations to create increasingly complex relationships between words or parts of a sentence in a document.

IV. Datasets

The first dataset that we used in the experiments to evaluate the ste2img encoding scheme is Text-Ferramenta, which consists of adverts collected from an Italian price comparison website. The dataset is composed of text description of adverts in Italian language from online and physical retailers. The description contains features and other technical information about the items on sale. This produced a dataset characterized by short texts and grammatically ill formed sentences, which made this dataset more compelling. We created the ground



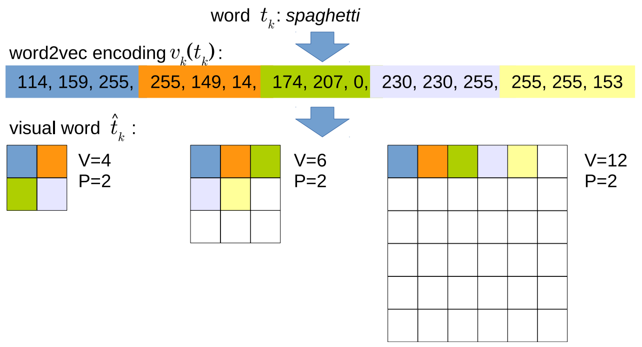


Fig. 3. A schematic example of our encoding scheme. The figure on the left shows a simple encoding image of size $W \times H = 9 \times 9$, with three visual words $\hat{t}_k$ of size $V \times V = 4 \times 4$ which may contains a maximum number of $(V/P)^2 = (4/2)^2 = 4$ superpixels of size $P \times P = 2 \times 2$. On the right, for example the word "*spaghetti*" is encoded in a sequence of 15 numbers using word2vec, and the same sequence can be transformed into different visual words by changing parameter V (in this example $V = 4, 6, 12$).

truth using a query based software that clusters commercial offers based on a text matching software. The dataset consists of 88,010 text description instances randomly split in 66,141 for train and 21,869 for test sets, belonging to 52 classes, e.g. paint brush, hinge, tape, safe, chain, ladder, cart etc. in a hardware category. Text description in dataset contain 22,045 different words for train and 20,083 for test sets. We will release this dataset with our work for research purposes.

In addition to the Text-Ferramenta dataset, we used two publicly available datasets i.e. *StackOverflow* and *20news* in experiments to validate our ste2img encoding scheme. StackOverflow dataset contains 20.000 question titles from 20 classes as described in [8]. Each class has 1.000 question titles, randomly split into train and test sets. We have 16.000 and 4.000 instances for train and test respectively. In our experiments, we only used question titles to train word embeddings. We downloaded the dataset from the github page[1].

The third dataset that we used in experiments is called 20news. The 20news dataset has become a popular choice for experiments in Natural Language Processing applications, such as text classification and text clustering. 20news dataset contains 20.000 newsgroup partitioned into 20 different classes. In our experiments, we used 20news-bydate version of the dataset[2]. This version contains separate train and test sets which are grouped into six major categories. We selected four major categories: comp, politics, rec, and religion as described in [9]. These four categories contain 7.977 and 5.321 instances for train and test respectively.

We converted the text from StackOverflow, 20news-bydate and Text-Ferramenta to lowercase, erased punctuation and special symbols from text before the encoding process. However, for the 20news-bydate dataset, we investigated the contribution of the preprocessing step in our approach for text classification. The summary statistics of the datasets are described in Table I.

V. Experiments

The aims of the experiments are:

[1]https://github.com/jacoxu/StackOverflow

[2]http://qwone.com/∼jason/20Newsgroups/

TABLE I

Statistics for text datasets. C: number of classes; Train/Test: number of instances for train/test and Vocab size: number of unique words in the dataset; Max words: the maximum number of words per document.

Dataset	C	Train	Test	Vocab size	Max words
Text-Ferramenta	52	66.141	21.869	163797	104
StackOverflow	20	16000	4000	30919	64
20news-bydate	4	7977	5321	234316	11278

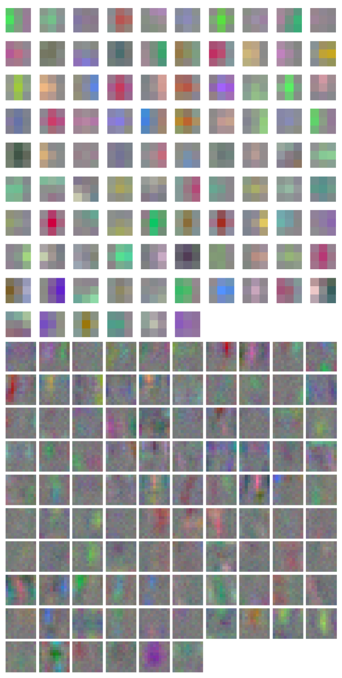


Fig. 4. Two CNN kernels of the "conv1" layer. On the top 96 kernels 3×3 while on the bottom the same number of kernels having size 11×11. The kernels were trained on 20news-bydate dataset using the following configuration shown in Fig. 5: Size of visual word = 24×24, num. MAX words = 80, superpixel size = 4×4, 12 word2vec features.

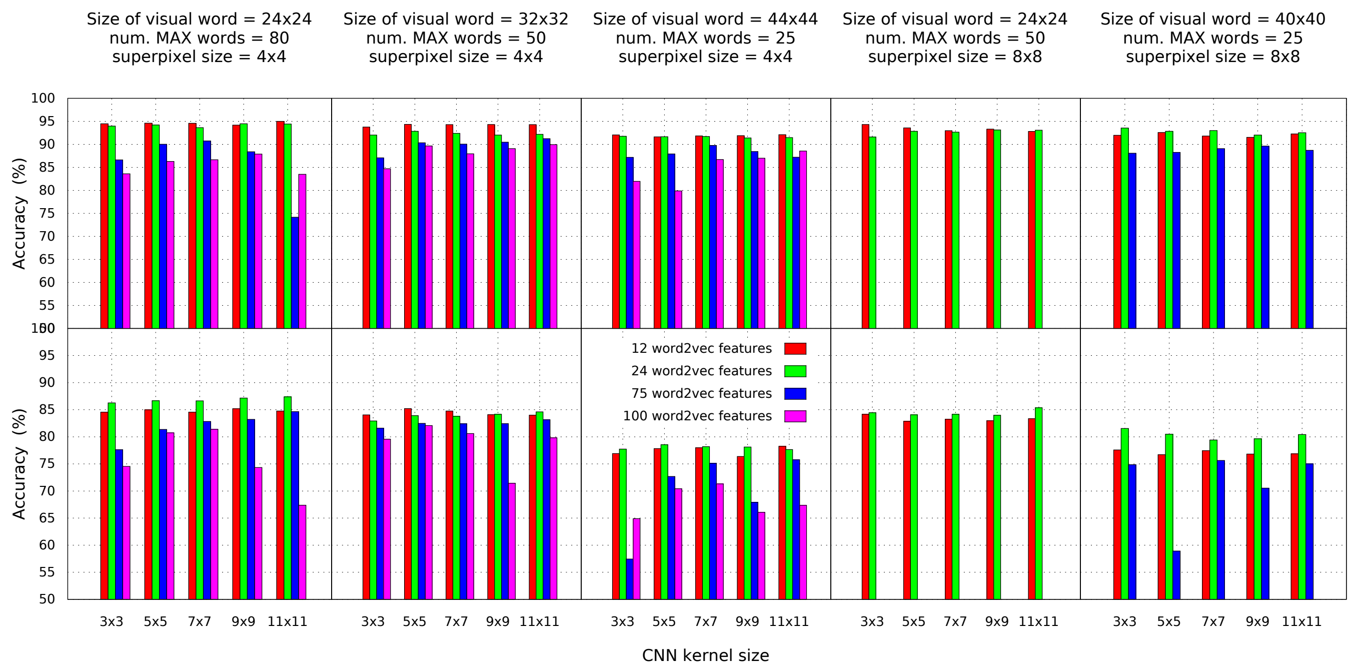


Fig. 5. Histograms of CNNs accuracies reached on 20news-bydate dataset when given the *ste2img* encoding in input. The upper row shows the results obtained from the original text transformed in lowercase and from which were removed all the characters different from [a-z] and [0-9]. The bottom row shows the results obtained from the original text. "kernel size" is the size of the convolution kernels for the input CNN layer called "conv1" in Fig. 2. "num. MAX words" is the maximum number of textual words that can be encoded within the image. "Word2Vec features" is the number of Word2Vec features used to encode a single textual word as a sequence of colored superpixel within a visual word.

- To validate the proposed encoding scheme and understand different parameters and configurations
- To compare the proposed encoding scheme with text classification based on Doc2Vec and SVM

In our experiments we measured the performance of models using overall classification accuracy. The encoding scheme mentioned in Section III produces encoded images from Word2Vec features. These encoded images can be classified using CNNs, however, we used the AlexNet architecture in our experiments. The architecture contains eight layers with weights; the first five are convolutional and the remaining three are fully-connected. The output of the last fully-connected layer is fed to softmax which produces a distribution over the class labels, as shown in Fig. 2.

We use the publicly available Doc2Vec tool to train word embeddings, and all parameters are set as in [1] to train sentence vectors on 20news-bydate and Text-Ferramenta datasets. However, we use a smaller window size (7), for StackOverflow dataset, as it is composed of short question titles. Normally, Doc2Vec and Word2Vec are trained on a large corpus and used in different contexts. However, in our work, we trained these two tools with the same training set for each dataset used for text classification.

We carried out experiments with different kernel size as shown in histograms of Figs. 5, 6, 7. All experiments are conducted using $s = 1$, images of 256×256 and we never removed stop words from the text. We used superpixel, described in Section III, set to 4×4 and 8×8 and we were interested in understanding if best results can be obtained by using a kernel size that is smaller or larger than superpixel. From histograms of Figs. 5, 6, 7 we concluded that in most cases different kernel size produce similar results. We can see that best results are obtained using a low number of features: 12 or 24. We believe that this resulting difference is due to the encoding technique. In fact for low number of features, our approach encodes feature vectors with superpixels in a single row, while, with large number of features, multi-line encoding is used. The impact is given by the convolutional layer, which cannot recognize scrolling images from left to right thus encoding in a new line as if it belonged to a previously seen word. This problem does not happen when features are encoded in a single line, because the network processes a whole word before analyzing the next and we know from Fig. 4 that different kernel size capture a single superpixel with its immediate neighbors. Some bars in the Figs. 5, 6, 7 are missing because of large number of features, and it is not possible to encode features in image due to larger size of superpixel.

We compared our encoding scheme with text classification based on Doc2Vec and SVM. We obtained feature vectors from Doc2Vec model for each instance starting from each document of a dataset and saved embedding to a file, one for training and one for testing instances. Finally, instances have been classified with SVM available in the WEKA open source library [10] to compare the results against ste2img as shown in table II. These results indicate that our encoding scheme

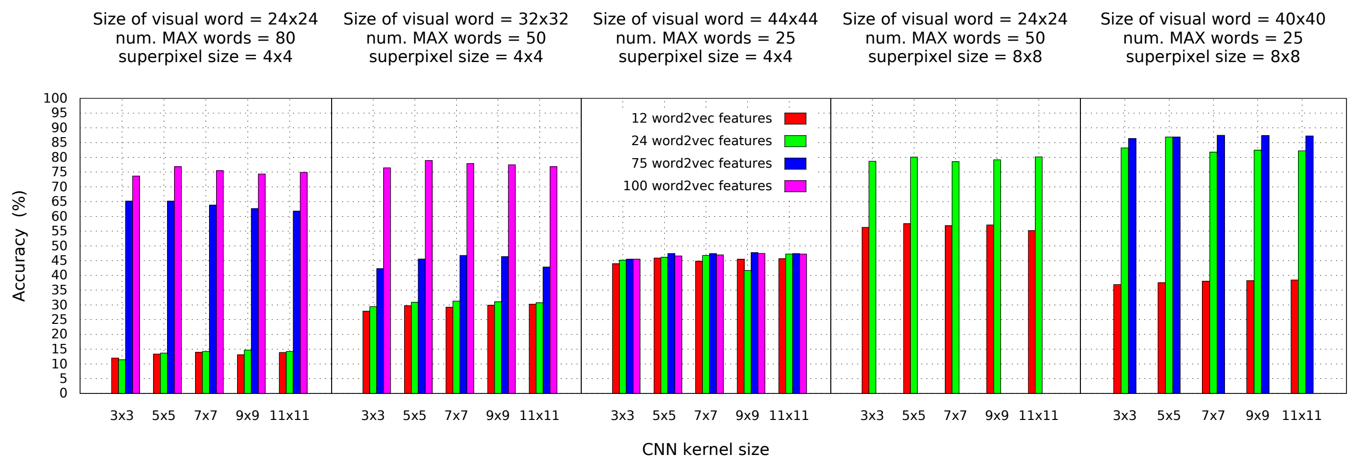


Fig. 6. Histograms of CNNs accuracies reached on stackoverflow dataset when receiving the *ste2img* encoding in input. "kernel size" is the size of the convolution kernels for the input CNN layer called "conv1" in Fig. 2. "num. MAX words" is the maximum number of textual words that can be encoded within the image. "Word2Vec features" is the number of Word2Vec features used to encode a single textual word as a sequence of colored superpixel within a visual word.

outperformed text classification based on Doc2Vec and SVM for StackOverflow and Text-Ferramenta datasets. However, 20news-bydate dataset does not produce similar results with our encoding scheme for 75 and 100 features, because we limit the maximum number of words for each document.

TABLE II
COMPARISON OF THE ACCURACY OF THE DOC2VEC USING SVM AND THE BEST RESULTS OBTAINED WITH OUR PROPOSED *ste2img*.

num. features:	12	24	75	100
SVM Text-Ferramenta	71.02	80.47	87.83	88.69
ste2img Text-Ferramenta	**89.90**	**91.60**	**91.84**	**89.38**
SVM StackOverflow	44.80	61.95	68.62	70.62
ste2img StackOverflow	**45.90**	**86.88**	**87.45**	**78.42**
SVM 20news-bydate	91.54	93.33	**95.30**	**95.88**
ste2img 20news-bydate	**94.99**	**94.48**	91.22	89.93

VI. CONCLUSION

In this work, we proposed a pipeline that can encode the features of Word2Vec words of a text document in an image, using the CNN capabilities to classify text document. Our proposed scheme outperformed text classification based on Doc2vec and SVM on three datasets i.e. StackOverflow, Text-Ferramenta and 20news-bydate. Our results indicate that the proposed scheme can be applied to any kind of sentences such as technical descriptions, common sentences or ill-formed phrases with different lengths. Finally, we believe that this work can be used in future multimodal approach leading to further results. Further investigations about number of encoded words per document are needed to better understand the relationships between encoding parameters and results. Furthermore, to avoid limitation of document length we want to study more compact embedding techniques like, for example, the sentence level embedding proposed in [11].

ACKNOWLEDGMENT

We also gratefully acknowledge the support of NVIDIA Corporation with the donation of the GeForce GTX 980 GPU used for this research.

REFERENCES

[1] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems*, ser. NIPS'13, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 3111–3119.

[2] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," *Machine learning: ECML-98*, pp. 137–142, 1998.

[3] M. Rogati and Y. Yang, "High-performing feature selection for text classification," in *Proceedings of the eleventh international conference on Information and knowledge management*. ACM, 2002, pp. 659–661.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[5] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.

[6] P. Blunsom, E. Grefenstette, and N. Kalchbrenner, "A convolutional neural network for modelling sentences," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, 2014.

[7] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin, "Learning sentiment-specific word embedding for twitter sentiment classification." in *ACL (1)*, 2014, pp. 1555–1565.

[8] J. Xu, P. Wang, G. Tian, B. Xu, J. Zhao, F. Wang, and H. Hao, "Short text clustering via convolutional neural networks," in *Proceedings of NAACL-HLT*, 2015, pp. 62–69.

[9] S. Hingmire, S. Chougule, G. K. Palshikar, and S. Chakraborti, "Document classification by topic labeling," in *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2013, pp. 877–880.

[10] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[11] V. Mijangos, G. Sierra, and A. Montes, "Sentence level matrix representation for document spectral clustering," *Pattern Recogn. Lett.*, vol. 85, no. C, pp. 29–34, 2017.

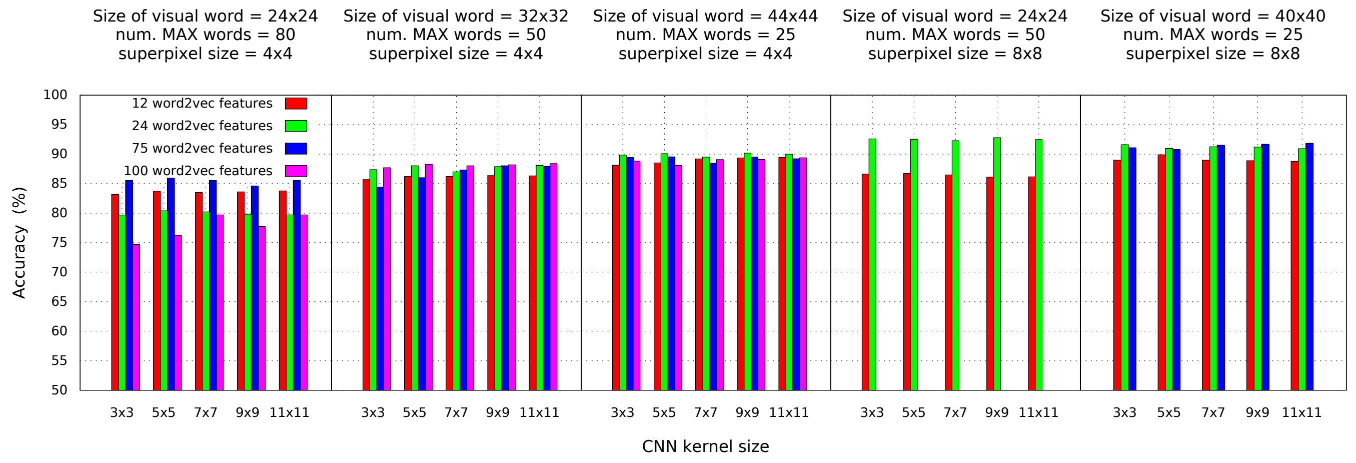


Fig. 7. Histograms of CNNs accuracies reached on Text-Ferramenta dataset when receiving the *ste2img* encoding in input. “kernel size” is the size of the convolution kernels for the input CNN layer called “conv1” in Fig. 2. “num. MAX words” is the maximum number of textual words that can be encoded within the image. “Word2Vec features” is the number of Word2Vec features used to encode a single textual word as a sequence of colored superpixel within a visual word.

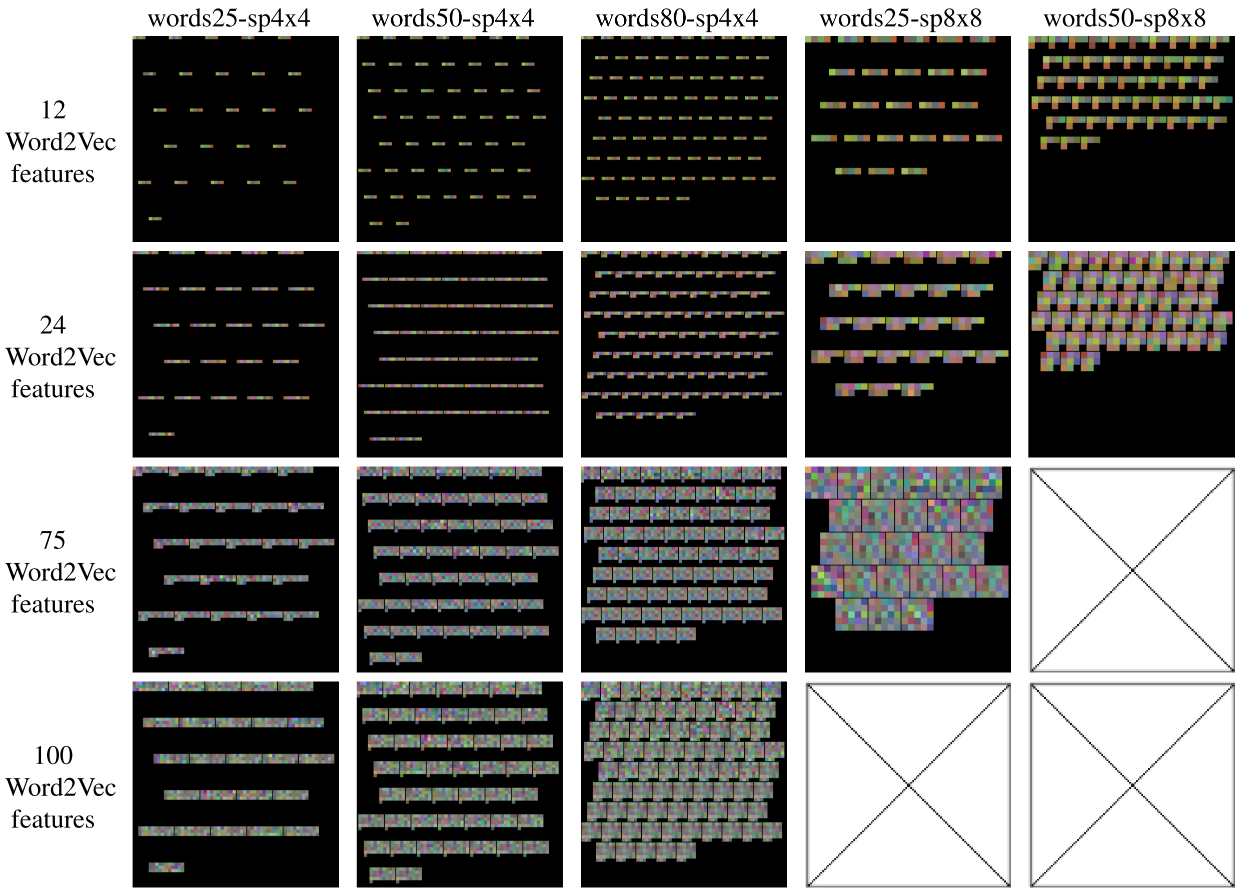


Fig. 8. Images encoding the same document “20news-bydate-train/comp.os.ms-windows.misc/9474” of the 20news-bydate dataset. “Word2Vec features” is the number of Word2Vec features used to encode a single textual word as a sequence of colored superpixel within a visual word. “words” is the maximum number of textual words that can be encoded within the image. “sp” is the size of each superpixel. The three empty cells correspond to configurations that can not be encoded in images of size 256 $\times$ 256.