Using Mixture Density Networks to Embed Images in Text Topic Spaces

Sergi Garcia

Abstract

The lack of adequate data to train deep learning networks in a supervised manner has led to an increased popularity of self-supervising learning methods. These methods rely on proxy tasks to provide the supervisory signal required to train the networks, and usually benefit from large, freely available sources of data. The work presented here is based on TextTopicNet, a self-supervised framework for learning semantic image representations, where the supervisory signal is text. In TextTopicNet, the network is trained to predict the embedding of any given image into a pre-calculated text topic space. We argue that teaching the network to predict a single embedding for an image offers a limited description power of its semantic content, since a single image can be found in diverse semantic contexts. Instead, we propose to learn for every image its conditional density function over the embedding space using Mixture Density Networks. We expect that this strategy will offer a more complete semantic description of the contents of the image, and therefore produce better features.

I. Introduction

One of the biggest challenges in computer vision and deep learning comes with collecting and manually annotating large datasets such as ImageNet [5] or MS COCO [26]. Self-supervised learning is an attempt to overcome these difficulties, and usually benefits from large and freely available sources of data. In self-supervised learning, the labels are obtained from mining the data and designing proxy tasks on it. These proxy tasks provide the supervisory signal required to learn the patters in the data. Examples of such tasks include colorization [22, 23], inpainting [32] or audio prediction [30, 2].

We have centered our attention on TextTopicNet [12, 31], a self-supervised learning framework where the supervisory signal comes from Wikipedia articles, a multi-modal source of information that comprises text and images. The network is trained by learning to predict the topics of the article, which acts as the supervisory task. By learning to predict the semantic context of the images, the convolutional neural network produces distinctive visual features. When the authors tested their framework in 2017, they achieved state-of-the-art results on retrieval and classification tasks.

In this work we argue that a single image can have a wide range of topics associated with it, and that by teaching the network to output a single embedding we are effectively averaging all the possible topics the image may be associated to. For example, an image of a building can be related to "architecture", but depending on the textual description it is attached to, it may belong to other topics such as "politics" or "religion" (if the building is for example a parliament or a church). Therefore, instead of outputting a single embedding for every image, we have decided to predict the probability density function over the topic space of every image. This way we are using a much more rich representation of the semantics of the image, and we expect that this will help to obtain better results in tasks such as retrieval and classification.

To model the probability density functions of the images over the topic space we have decided to use a Mixture Density Network (MDN) [3]. The MDN is a model that learns to predict a mixture of Gaussian kernels that represent the conditional probability density function over the target space, conditioned by the input. The architecture consists of a generic feed-forward neural network (in our case a convolutional neural network) coupled with a mixture model. The mixture model outputs the parameters of mixture, and the loss for every image is the negative log-likelihood of the textual embedding associated with the image. We can see an overview of our framework in the figure 4.

To evaluate the quality of the learnt features we will test our framework on classification and retrieval tasks. These were the same tasks used to evaluate TextTopicNet, this way we will be able to directly compare both approaches. We will also rely on the visualization of the embedding space to understand whether our strategy is working or not. First we will overview some work related to this project such as text embedding and topic modeling frameworks, self-supervised learning and or joint text-image embedding frameworks. We also explain more in depth the how TextTopicNet and Mixture Density Network work. In the next section we describe our architecture, explain the proxy tasks we performed and present the results obtained. Having seen the results, we offer a general analysis of the results and end up with the conclusions of the project.

II. Related Work

In this section we are going to review some of the work related to our project. Since we are using joint text and image embeddings to perform self-supervised learning, we have included works on text embeddings and topic modeling, self-supervised learning and some articles that use joint image-text embeddings.

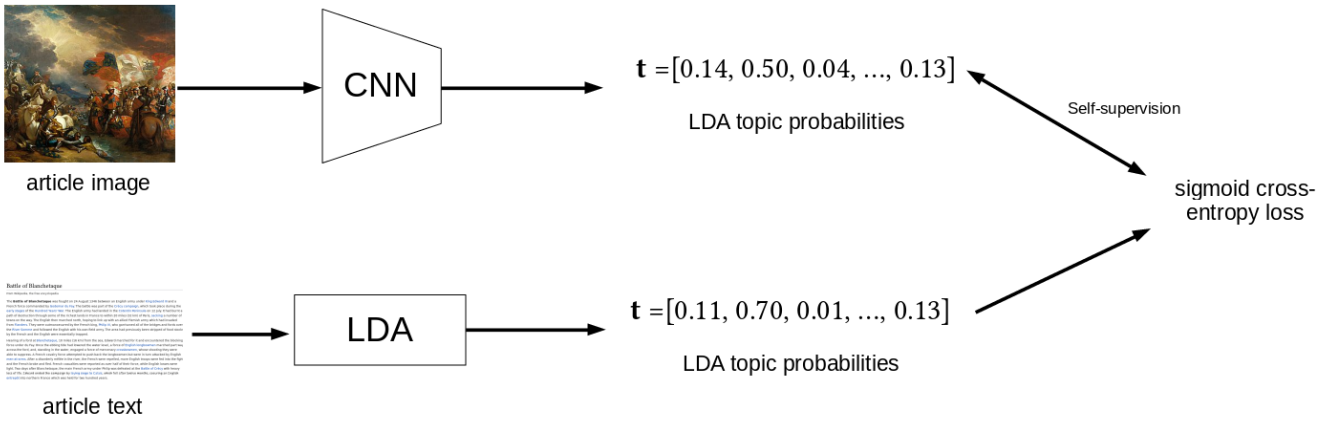


Fig. 1: Overview of the TextTopicNet framework. The data comes from articles of Wikipedia. The CNN has to generate a vectorial representation of the topics of the image. The LDA embedding of the article text serves as the supervisory signal.

A. Topic Modeling and Text Embedding

One of the most well known text embedding frameworks is Word2vec [29], a model used to generate vector representations of words. Word2vec takes a corpus of text in form of one-hot encoding and outputs a vector space of lower dimensionality. The authors presented two ways of learning the representation, a continuous bag-of-words model and a skip-gram model. The bag-of-words model learns by predicting the current word from the representation of the surrounding words, which are either averaged or concatenated (it does not take into account the order of the words). The skip-gram model learns by predicting the context of the words surrounding the current word. This way, the model is capable of capturing semantic meanings of the words, and similar words fall closer in the embedding space.

GLoVE (Global Vectors for Word Representation)[33] is a combination of Word2vec and Latent Semantic Analysis (LSA) techniques [21]. LSA uses matrix factorization techniques, which are good at capturing the global text statistics, while Word2vec and other similar methods are good at describing the semantics of words. GLoVE benefits from both approaches to produce better word embeddings.

When we need to produce fixed length representations of texts of variable length, one of the most common approaches is bag-of-words. The main drawbacks of this method is that the order of words is lost and that their semantics are ignored. Doc2vec [24] appeared as an alternative to these methods. This framework is very similar to methods like Word2vec, and the main change is the addition of a token at the beginning of each paragraph, which is mapped to a vector using different weights than the words. This token acts as a memory of the rest of the words of the paragraph, and describes its general topic.

In our case, we are going to perform the embedding between the image and the text of the articles in the Latent Dirichlet allocation (LDA) [4] topic space, since the authors of TextTopicNet obtained better results using this representation. LDA describes each document as a mixture of the probabilities of learnt abstract topics. Each one of these topics is described by its probability distribution over the words of the corpus.

B. Text-Image embeddings

Like in TextTopicNet, our method relies on joint image and text embeddings as the auxiliary task to learn the image representation. There are several previous contributions that benefit from embedding text and space in a common space.

In the DeViSE [10] framework, the authors train separately a neural language model (Word2Vec) and a neural model for image recognition (Alexnet [20]). Then, they take the already trained layers from the CNN and fine-tune them to predict the vector representation of the image label text. They use a similarity metric between the vector outputted by the CNN and the vector outputted by the language model.

In [39] they propose a method for learning joint embeddings of images and text using two-branch neural networks. Given a training image, the objective is to have a small distance between the image and the matching sentences, while with non matching sentences the distance has to be bigger. They claim that their architecture achieves state of the art results in image-to-text and text-to-image retrieval. Like in DeViSE, they used Word2Vec to model the joint embedding space.

Salvador et al. [35] introduce Recipe1M, a dataset composed of recipes and food images. They use joint text-image embeddings to perform retrieval of images and recipes. Their architecture encodes the textual information using LSTM cells [16], while the image is embedded using the VGG-16 [36] or the ResNet-50 [14]. They train using negative and positive pairs in combination with a cosine similarity loss (they also use a semantic regularization loss).

Other related works in this topic include [13], where image captions are used to learn a semantic-aware visual representation. Using this representation they tackle the problem of image retrieval in complex scenes, where the goal is to retrieve semantically similar images from a query image.

Our work differs from the previous contributions in that we are not modeling the semantics of the image with a single vector, but rather predicting conditional density function of the topics of the image.

C. Self-supervised learning

Due to the difficulty to obtain manually large-scaled annotated datasets, there has been an increasing interest on self-supervised learning methods. These methods often rely on non-visual signals to learn the representation of the visual information. These signals are correlated to the images, and offer a way to self-supervise the learning of the visua representation. For example, the authors in [6] train a network to predict the relative position of an image patch given another patch of the same image. The authors argue that by training a network on this task, it is forced to recognize the objects of the pairs.

In [40], videos are used to self-supervise the training of the network. The idea is that two patches connected by a track should have a similar visual representation, since they belong to the same object. A Siamese-triplet network with a ranking loss function is used to train the CNN representation. When this work was published, it obtained results close to the state of the art (including fully supervised methods) in the VOC 2007 classification challenge [9]. Lee et al. [25] also use videos, but in this case the authors use the statistical temporal structures of the different frames as the supervisory signal. Given a sequence of unsorted frames sampled from a video, the network has to learn to sort the frames by chronological order.

The authors in [1] use egomotion information (relative movement) obtained by sensors mounted on a vehicle. The authors draw inspiration from biology, where living organisms develop their visual abilities for the purpose of moving and acting in the world. They train a network to predict the camera transformations between two image pairs using a contrastive loss. A similar idea was also proposed by [17]. In relation with motion, Mahendran et al. [28] use optical flow as the auxiliary task. The authors present a method that computes per-pixel embeddings in such way that the similarity between pixel embeddings matches the similarity between their optical flow vectors.

Pathak et al. [32] take inspiration from auto-encoders. The authors propose a Context Encoder where a CNN is trained to generate the contents of a image region conditioned on its surroundings (this task is also known as inpainting). The authors affirm that, in order to inpaint the missing area of the image, the network has to understand the content of the entire image and, at the same time, produce a plausible hypothesis for the missing parts. The authors observed that using a reconstruction plus an adversarial loss produced better results than just only using a pixel-wise loss.

A fully adversarial approach was proposed by Donahue et al. [8]. The authors argue that the semantic information learnt by the adversarial models can be useful in auxiliary problems where semantics are relevant, but they provide no way of learning the inverse mapping (project the information back to the latent space). They propose Bidirectional Generative Adversarial Networks (BiGANs) to learn this inverse mapping, and they affirm that the learnt features are useful for supervised tasks, matching the results by other unsupervised and self-supervised methods.

Other modalities such as audio have been explored as supervisory signals [30, 2]. Owens et al. [30] train a convolutional neural network to predict a statistical summary of the sounds associated with a video frame. Using sound as a supervisory signal, the network is capable of identifying the visual representation of different scenes, and achieve state-of-the-art on recognition tasks among other self-supervised methods. Arandjelovic and Zisserman [2] remark the potential amount of data available in videos in terms of both images and audio, and how it can be used to train and self-supervise convolutional neural networks. The authors use a simple binary classification task (which they name *audio-visual correspondence task*) as their proxy task. Given a pair of image and audio signals, the network has to decide if they correspond each other or not. The network contains a vision subnetwork, which processes the image, and an audio subnetwork, which receives a 1-second audio clip. The output of both networks is coupled and then the network decides if both signals correspond to each other.

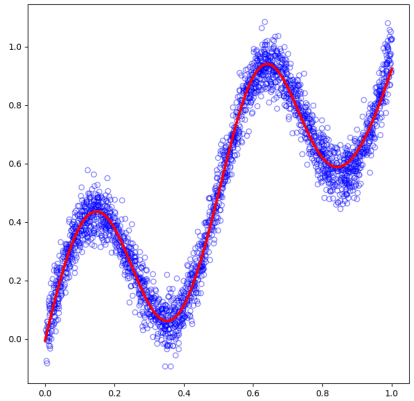
There are many other modalities that have been explored such as colorization [22, 23], which takes a grayscale image as input. Doersch and Zisserman [7] experiment with different techniques of self-supervised learning, and they train using different combinations of these methods. They conclude that combining tasks and using deeper networks (they trained using ResNet [14]) always improves the performance of the networks.

More recently, Hénaff et al. [15] tackles self-supervised learning with Contrastive Predictive Coding, a self-supervised objective that learns from sequential data by predicting the representations of future observations from those past ones. Trinh et al [37] also use Contrastive Predictive Coding, but in this case the network is given masked-out patches of an input image, and it has to learn to select the correct patch among other sampled from the same image to fill the masked location.

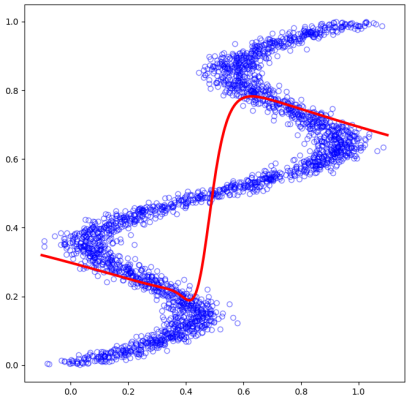
From all the presented works, our approach is more closely tied to the approach proposed by [12, 31], where the authors use textual embeddings as the supervisory signal. We explain this framework more in depth in the following section.

D. TextTopicNet

The work behind this project has been essentially based on the TextTopicNet framework [12, 31]. Like in the previously described contributions, the authors designed an auxiliary task as a method for self-supervision in order to learn visual features.



(a) Regression problems with one target for each input are easily solved using the sum-of-squares error function.



(b) When the target space is multi-valued, the sum of squares strategy is not suitable. The model learns to output the average of the target space for each input, and many of the output values are not correct.

Fig. 2: Comparison between single valued and multi-valued output spaces.

This self-supervision is achieved by using multi-modal information, in particular articles with text related to the image used to ilustrate it. The visual features are learnt by training a CNN to predict the most likely semantic context of a given an image. This semantic context is modeled using text embeddings, which act as the labels for every image and supervise the training. The chosen text embedding model is the Latent Dirichlet Allocation [4]. In [31] the authors compared other textual embedding approaches such as Doc2Vec [24] or GloVe [33], and showed that LDA gives better image classification performance than the other embedding techniques. They achieved state-of-the-art results in image classification, multi-modal image retrieval and object detection in the context of self-supervised learning methods.

The TextTopicNet architecture is based on the CNN CaffeNet [18], a replica of AlexNet [20] with some differences. The reason behind using AlexNet is to have a point of comparison with the other self-supervised training methods. They also provide a smaller version of the network used to do experiments with tiny images. The output of the network has as many outputs as the number topics of the LDA space (40 topics in the reported implementation, although they experiment with different number of topics), and it is followed by a sigmoid activation function. The choice of the number of topics was based on the validation accuracy on the PASCAL VOC2007 [9] dataset for different number of topics. The objective function of the network is to minimize the sigmoid cross-entropy loss between the output of the network for a given image and the corresponding LDA embedding of the article it belongs to. Only the English portion of the Wikipedia ImageCLEF [38] is considered for training the network.

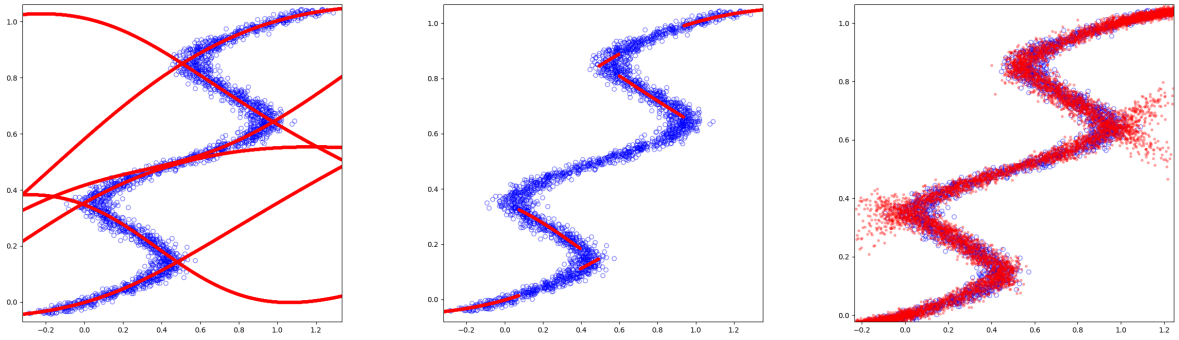
The usual approach to evaluate self-supervision frameworks is to perform auxiliary tasks on the trained network. In [12] they performed classification on the Pascal VOC2009 [9] and retrieval on a multi-modal (image-text) Wikipedia dataset [34]. To perform classification they extracted the features from the last layer of CaffeNet (they also report results on other shallower layers) and classified using a linear SVM for each class of the dataset. In retrieval they compared the image embeddings with the textual embeddings of the articles using the Kullback-Leibler divergence. We will perform the same tasks to directly compare TextTopicNet with our approach.

E. Mixture Density Networks

Mixture Density Networks (MDN) were introduced by Bishop [3]. In the original work the author addresses the problem of the limited power of description of the sum of squares and cross entropy error functions in the context of regression and classification. These strategies are not suitable when the output data is multi-valued, since these error functions train the model to output the conditional average of the target data (conditioned by the input vector).

One example of this can be seen in figure 2. In the case of figure 2a, the regression problem is easily solved by building a simple neural network that uses the sum-of-squares error function to optimize the model. When we are presented with the inverse problem though (figure 2b), we can clearly see that this strategy is not adequate.

MDNs were introduced to solve this problem. As the author demonstrates, the least-of-squares and the cross entropy formalisms approximate the conditional average of the target data. Assuming that the conditional distribution of the data can be modeled as a Gaussian function, the author proves how we can recover both formalisms using maximum likelihood. This motivates the idea to extend the single Gaussian approach to a mixture of Gaussians, which is intended to express the whole conditional probability density function over the target space $p(t|x)$, where t represents the target space and x the input space. The network outputs the various parameters of a mixture of Gaussians conditioned by the input vector x, which are the mixing coefficients, the means and the variances of the different kernels. These outputs are preceded by a conventional



(a) Means of the 6 kernels produced by the network for different values from -0.2 to 1.2.

(b) In this figure, the values shown come from the center of the kernels with the highest mixing coefficient.

(c) Examples randomly sampled from the mixtures generated by the network.

Fig. 3: Result of applying a Mixture Density Network to the problem presented in 2b with a number of 6 kernels.

feed-forward neural network, and the combined structure is called a Mixture Density Network. The weights of the network are adjusted by minimizing the negative log-likelihood between the distribution produced by the network and the data.

Applying a MDN to the previous regression problem (figure 2b) results to a model that outputs a mixture of Gaussians that expresses the conditional probability over the target space (y axis) conditioned by the input (x axis). Figure 3a shows the average values of the different Gaussian kernels given different inputs over an interval from -0.2 to 1.2. As it can be seen, the network has been able to adjust the Gaussian kernels to the manifold of the target data. In the figure 3b we can see the center of the kernel with the biggest mixing coefficient over the same interval. In the figure 3c we can see how we can randomly sample from the mixture model and approximate the training data.

The paper offers a few options to find the maximum of the probability density function generated by the network. Finding the maximum of a mixture of Gaussians is a non linear optimization problem, but there are other options. If the kernels are not heavily overlapping, one option can be selecting the center of the kernel with the highest component, taking into account its mixing coefficient and the covariance of the kernel. Another strategy is to consider the density mass of each kernel.

III. MDNs for Text-Image embeddings

In this project we want to combine the power of description of Mixture Density Networks with the TextTopicNet framework. In TextTopicNet the network generates a single vector that describes the topics of an image. Our reasoning is that the same image can in principle be used to illustrate different articles, hence it should be mapped in a number of different regions of the topic space at the same time. Training the network to output a single topic vector can lead the model to average the topics that this image may belong to. Instead of outputting a single topic vector, we want the model to output a whole probability density function over the topic space by using Mixture Density Networks. We theorize that, as the model is generating a much richer description of the contents of the images, the performance of the network in proxy tasks such as classification and retrieval will be improved. We also expect that the visualization of the embedding space will also show better defined clusters of images.

Regarding the visualization of the embedding space, in the visualization of TextTopicNet (figure 5) we observe that many images are mapped at in the center of the space. We consider that these images belong to various topics, and that these topics have been averaged and therefore fall in the center of the image. A second way to see this is to check the topic distribution of the space: the topics of the images in the center tends to be quite uniform, which indicates the averaging of various topics. On the other hand, the images at the end tend to have more sparse topic distributions. We expect that these images will be split in different clusters belonging to different components of the mixture, as they may be associated with different topics.

A. Architecture

Our network architecture follows the same design as described in the MDN paper [3], a feed-forward neural network followed by a mixture model, which outputs the probability density of the target data $p(\mathbf{t}|\mathbf{x})$. This density function is expressed as a combination of Gaussian kernels:

$$p(\mathbf{t}|\mathbf{x}) = \sum_{i=1}^{m} \alpha_i(\mathbf{x})\phi_i(\mathbf{t}|\mathbf{x}) \tag{1}$$

where m is the number of kernels of the mixture, $\alpha_i(\mathbf{x})$ is the mixing coefficient of the i^{th} kernel of the mixture, and expresses its contribution to the density function. The mixing coefficients could be seen as the prior probability of each kernel,

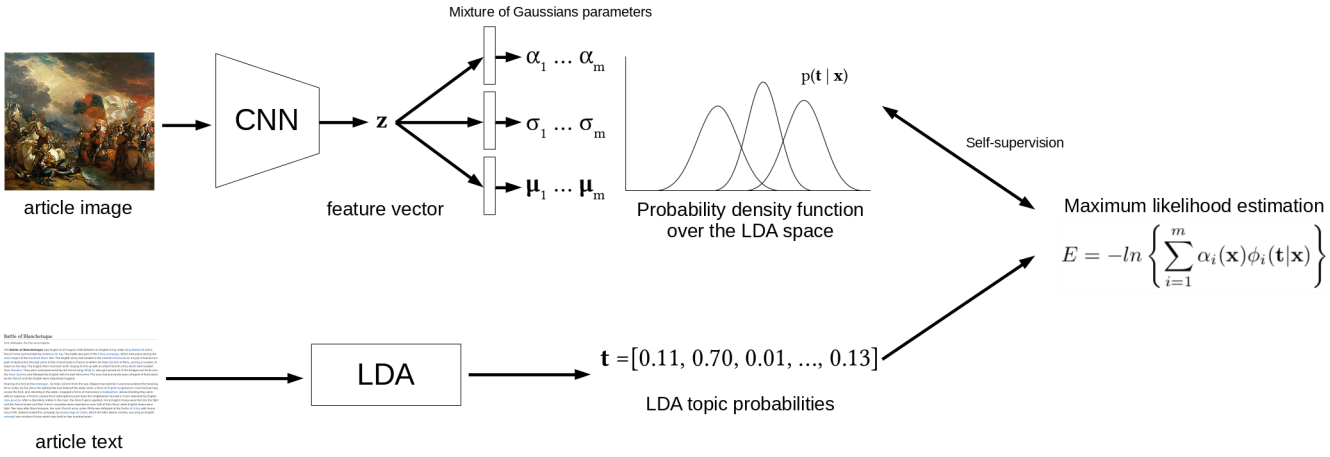


Fig. 4: Like in TextTopicNet, we use pairs of image and text extracted from articles from the English Wikipedia to self-supervise the training of the network. We extract the topics of the articles using Latent Dirichlet Allocation [4], which outputs a vector of probabilities of learnt abstract topics. Using a CNN in combination with three fully connected layers (which together form a Mixture Density Network), the network outputs the parameters of a mixture of Gaussians: the mixing coefficients α_i, the covariances σ_i and the means $\boldsymbol{\mu_i}$ of each component. This mixture represents the probability density function of possible topics of an image over the LDA space. We use maximum likelihood estimation to adjust the weights of the network.

and are constrained to $\sum_{i=1}^{m} \alpha_i(\mathbf{x}) = 1$. $\phi_i(\mathbf{t}|\mathbf{x})$ expresses the conditional density of the i^{th} kernel. The kernels are Gaussian functions:

$$\phi_i(\mathbf{t}|\mathbf{x}) = \frac{1}{(2\pi)^{c/2}\sigma_i(\boldsymbol{x})^c} e^{-\frac{1}{2}\frac{\|\boldsymbol{t}-\boldsymbol{\mu}_i(x)\|^2}{2\sigma_i(x)^2}} \tag{2}$$

In the previous expression, $\boldsymbol{\mu}_i(x)$ is the center of the i^{th} kernel, and c is the dimensionality of the target space, in our case the number of topics of the LDA. $\sigma_i(x)$ represents the variance of the kernel. We only define a single variance value for the kernel, instead of a full covariance matrix, resulting to isotropic Gaussians. This simplifies the architecture and the training of the model while it does not represent a limitation to the descriptive power of the mixture model, since we can approximate any density function given enough components and the right parameters of mean and variance [11].

The three components that define the mixture of Gaussians are functions of $\boldsymbol{x}$, and are the mixing coefficients α_i, the means $\boldsymbol{\mu}_i$ and the covariances σ_i. These parameters are modeled using the feed-forward neural network as previously mentioned, which takes $\boldsymbol{x}$ as the input and it is parameterized by its weights. In our case, this feed-forward neural network will be a convolutional neural network, which will output a feature vector $\boldsymbol{z_i}$ followed by a ReLU activation layer. The feature vector outputted by the CNN is the input for three fully connected layers that will output the three parameters of the mixture. For the mixing coefficients α_i, we have fully connected layer with m outputs (one mixing coefficient for every kernel). Since the mixing coefficients are constrained to $\sum_{i=1}^{m} \alpha_i(\mathbf{x}) = 1$, the output of this layer will be followed by a softmax activation function that ensures that they lay between (0, 1) and sum up to 1. For the convariances σ_i, the fully connected layer has also m outputs, since all the dimensions in one kernel share the same variance. In this case it is followed by an exponential activation function that ensures that the covariances are positive. Finally, the centers of the Gaussians $\boldsymbol{\mu}_i$ are computed with a fully connected layer with $m * c$ outputs (that is, m centers of c dimensions each one) without any kind of activation.

With all the components formed, we can compute the probability density function over the LDA space of an image $\boldsymbol{x}$. During training, we want to maximize the likelihood of the corresponding textual embedding of a given image. Therefore, the error of the network for any given pair of image $\boldsymbol{x}^q$ and LDA embedding $\boldsymbol{t}^q$ is defined as the negative log-likelihood:

$$E^q = -ln\left\{\sum_{i=1}^{m} \alpha_i(\mathbf{x}^q)\phi_i(\mathbf{t}^q|\mathbf{x}^q)\right\} \tag{3}$$

In order to backpropagate the error, the error of all the pairs of a batch is summed and averaged. We can see an overview of the framework in figure 4.

IV. Experiments

Since there is no direct way of evaluating the performance of the architecture, we performed a series of experiments that serve as proxy tasks. We include the same tasks used in TextTopicNet which are classification and retrieval, so that we can

Fig. 5: Mosaic generated with the vectors generated by TextTopicNet, reducing the 40-dimensions of each embedding down to 2 using t-SNE [27]

make a direct comparison with our framework. These tasks are also commonly used in other self-supervised works to have a common baseline between methods. Aside from classification and retrieval, we also include several visualizations of the embedding space and its properties that will help us understand how our network is learning. For comparison purposes, in the figure 7 we have included the visualization of some embedded images using the original TextTopicNet.

The first of the tasks is classification on the Pascal VOC2007 [9] dataset. In this dataset there are twenty object classes labeled, which include animals, vehicles and indoor objects. In order to evaluate the network on classification, we follow the standard practice of extracting a feature vector for each image from one of the layers of the convolutional neural network. With this feature vector we train a one vs. all linear SVM for each one of the classes of the dataset. In all the cases we selected the features produced by last layer of the convolutional network.

In the retrieval tasks we use the Wikipedia retrieval dataset from [34]. In this dataset we can perform image to text and text to image retrieval. When we are performing image to text retrieval we return the closest articles for a given image, and when we are performing text to image we return the closest images given a text document. The objective is to return the text or the images that have the same semantic label as the query. Both the text and the images belong to articles from the English Wikipedia. Like in TextTopicNet, the articles are embedded into the LDA space, but since the MDN network does not directly output an LDA vector for the images we cannot directly use any standard distance calculation between an article and an image. Instead we tried different strategies to find the closest images given a textual embedding or vice versa:

- Kullback-Leibler divergence: we tried computing the entropy between the textual embedding of an article and each one of the means of the mixture ($\boldsymbol{\mu_i}$) outputted by the network using the Kullback-Leibler divergence. This way, for a given pair of image and text, we have a list of entropy values, one for each center of the mixture. The center returned is the one associated with the minimum of all the computed distances.
- Euclidean distance: we also tried using the euclidean distance like in the previous point. We compute the euclidean distance between the textual embedding and each one of the centers of the kernels of the mixture, returning the lowest value. In general, the euclidean distance seemed to give better results than the Kullback-Leibler divergence.
- Likelihood estimation: the proper way of finding the similarity between a text embedding and the density function for a given image is to compute the likelihood of the textual embedding given the density function. If for example we have a textual embedding $\boldsymbol{t}_q$ and an image $\boldsymbol{x}_q$, we evaluate the density function (which is conditioned by $\boldsymbol{x}_q$) as follows:

Fig. 6: This figure shows a mosaic with the corresponding images placed at their respective kernel centers.

$$\mathcal{L}(\mathbf{t}_q|\mathbf{x}_q) = \sum_{i=1}^{m} \alpha_i(\mathbf{x}_q)\phi_i(\mathbf{t}_q|\mathbf{x}_q) \tag{4}$$

This way we directly obtain the likelihood of the text embedding given the probability density function of the image.

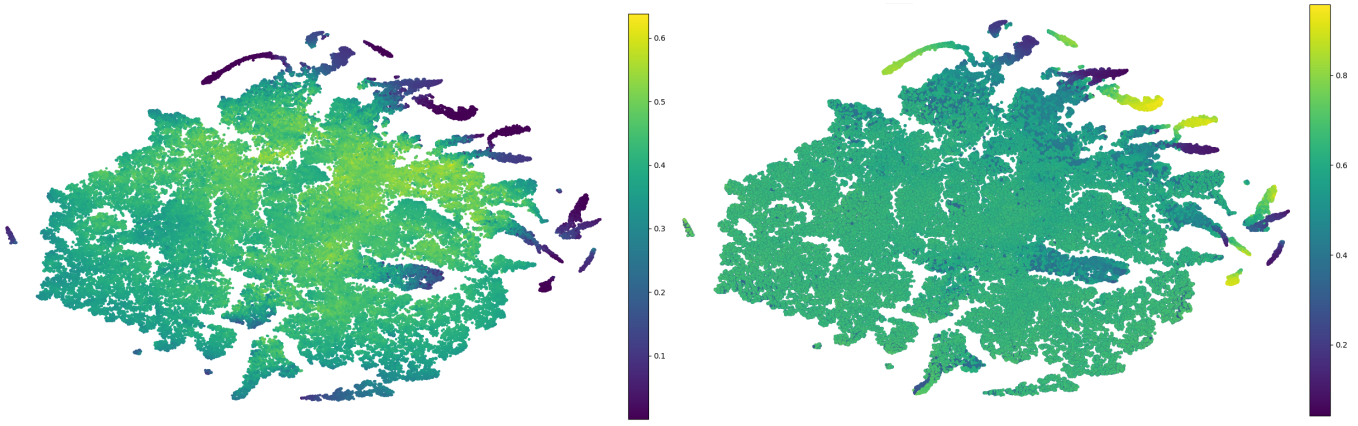
In order to compare our framework with TextTopicNet we have used Alexnet [20] as the backbone and we trained on the Wikipedia ImageCLEF [38], similarly to the original TextTopicNet. In the following section we will present and discuss the results using the previous framework, while in subsequent sections we explore other backbone architectures and larger datasets. Intermediate results are presented in the corresponding sections, while all the results obtained are summarised in section IV-D.

A. Base framework (Alexnet + ImageCLEF Wikipedia dataset)

The base framework uses an Alexnet backbone and is trained on the ImageCLEF Wikipedia dataset. We used the same subset used to train TextTopicNet, where only considerde English articles and filtered out the images that were not in JPG format or smaller than 256 pixels. The resulting datastet contains around 100k images that belong to 35k articles. The images have been previously resized to 256x256 pixels, and we perform data augmentation with a random horizontal flip and a random crop of 227x227.

To train the network we used stochastic gradient descent with a learning rate of 0.025, a momentum of 0.9 and a batch size of 64 images. The model trains for a total of 60 epochs, and the learning rate is multiplied by 0.1 every 20 epochs. We played with the number of kernels to see how the results evolved. We also tried changing the original number of 40 topics of the LDA space to 10 and 100.

1) Visualization: We will begin by visualizing the embedding space to analyze and understand if and how the network is learning. For this visualization we used a simpler mixture density network of 2 kernels. We computed the embedding of 100.000 images, and as a result we have 100.000 mixtures in the embedding space. We have decided that using two kernels



(a) In this figure, the intensity of the color of every point represents the value of the variance (the spread) of all the kernels outputted by the MDN. Higher values are represented in yellow and lower in blue. As expected, the spread of the kernels that are closer to the center is higher, and the kernels in the periphery have less variance, meaning that the network is more confident with these images.

(b) In this case the color represents the value of the mixing coefficients of every kernel. Colors close to yellow mean higher contribution to the mixture, while colors closer to blue low contribution.

Fig. 7: Visualization of the values of the mixing coefficients and variances for 100000 images. For every image, the MDN outputted a mixture of 2 Gaussian kernels. Every point in the image represents one of the centers of every mixture outputted by the network. The original embedding space had 40 dimensions, we reduced the dimensionality of the space using t-SNE [27].

would make the visualization easier, although we have to assume that we would observe similar results with more kernels. To reduce the dimensionality from 40 dimensions to 2 we used T-distributed Stochastic Neighbor Embedding (t-SNE) [27], a non-linear dimensionality reduction algorithm that is suitable for visualizing high dimensionality data. Since t-SNE learns a non-parametric mapping, we are not going to be able to visualize a heatmap of the density functions.

In figure 7 we can see the values of the variance and the mixing coefficients for each one of the centers of all the mixtures. In figure 6 we can see a visualization of the embedding space. For visualization purposes, every image is placed at the mean of each kernel of its mixture density function. In this case where we have two kernels, these would correspond to the two maximums of the density function.

The idea is that the t-SNE will reveal the different semantic clusters of images created by the network (assuming that each kernel of the mixtures corresponds to a semantic cluster). We expect that in well defined clusters, the variance of the kernels will be low. Our interpretation is that in these clusters the network has less trouble discriminating the topics of these images from the rest, and therefore the spread of the kernels is low. Higher variance in the kernels is expected where the clusters are less well defined, meaning that the network has more trouble modeling the semantic meaning of those images. In the figure 7a we can see the variance of each one of the kernels. In the periphery of the figure we have well defined clusters, with low variance, while in the center the variance of the kernels is higher, meaning that the network is less confident with these images. We observe a number of well defined clusters at the top and at the right of the embedding space which appear to have lower variance than the rest of the kernels. These clusters belong to classes such as vehicles, people or animals. TextTopicNet managed to create similar semantic clusters in the embedding space, as seen in the figure 5.

Analyzing the mixing coefficients (figure 7b) we can see that for the most part, all the kernels seem to have the same contribution to the mixture, aside from the well defined clusters at the top and the right. These clusters seem to come in pairs (the images in them have similar semantics), and one of the groups of kernels have a high contribution to their mixture while the other kernels have low contribution. In figure 8 we zoomed in one of these pair of clusters where the topic seems to be sports. In these two clusters the variance seems to be low (figure 8a), but the contribution of one of them seems to be higher than the other (figure 8b). These two clusters seem to contain the same images. This can be observed by plotting a line between the means of the two kernels for each image as seen in figure 9b. In figure 9a we use the same idea but with all the mixtures of the embedding space. As in the previous case, every pair of semantically related clusters contain the same images. As for the rest of the embedding space, in most cases both clusters fall on almost the same point.

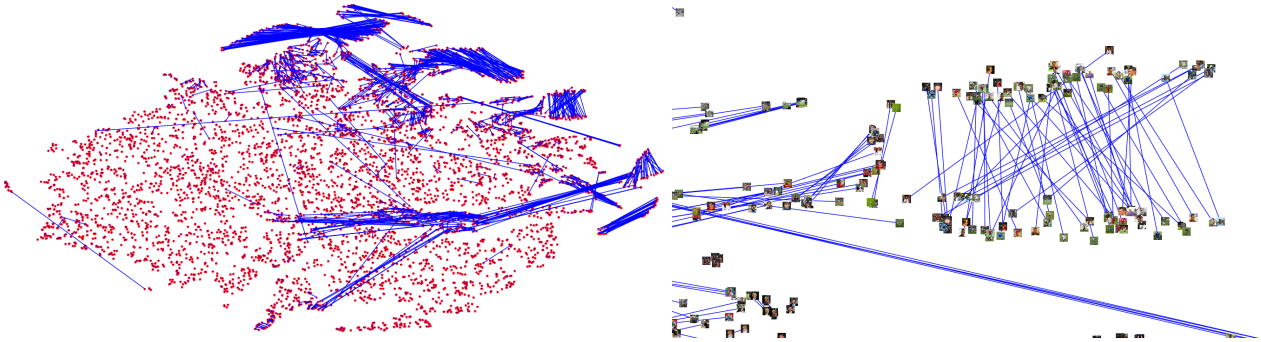
Now we can safely say that the model has managed to identify the different areas of the topic space for the same image. The downside is that we have not observed the network to better embed the images that were also difficult to embed using the original TextTopicNet. Instead, we have observed the opposite phenomenon, where the network expressive power has been used to embed images that TextTopicNet had less trouble embedding. Although we have used the 2 kernel case for this discussion, the conclusions are the same for the different combinations of number of kernels and topics.

(a) Variance of the kernels.

(b) Mixing coefficients of the kernels.

(c) Scattered images on top of their respective kernel centers.

Fig. 8: Detail of a pair of clusters with similar semantics. One of the clusters appears to have a high contribution to the mixture, while the other has a lower mixing coefficient. In both cases, the spread of the kernels is low.



(a) In this figure every red point is a center of a kernel, and the blue lines match the two kernels of the same mixture.

(b) In this case we have zoomed in the same region as in the figure 8. We can see how these two clusters contain the same images.

Fig. 9: These two figures show the two kernels of the the mixtures of the embedding space connected with a blue line.

2) Retrieval: In the task of retrieval on the Wikipedia dataset, TextTopicNet obtained a mAP of 0.36 in text to image retrieval and 0.4 in image to text retrieval. In table I, we can see how our architecture does not produce good results using the Kullback-Leibler divergence as previously explained. When changing the distance to Euclidean we can see that the results improve, but are still lower than in TextTopicNet. When we compute the likelihood (as described in 4), we obtain slightly lower results with respect the Euclidean distance. We seem to be obtaining higher scores in text to image retrieval, in contrast to TextTopicNet. Changing the number of kernels does not seem to improve the results, and we even appear to be obtaining slightly lower scores with more kernels. The highest score obtained in text to image retrieval is 0.33 with 3 kernels, and 0.26 in image to text with the same number of kernels, both obtained using Euclidean distance.

In the figure 10 we can see the top 5 images returned for text of topics ”warfare”, ”art”, ”biology”, and ”geography”. The images returned usually have a similar semantic topic to the text query, but in some occasions we can observe images that have a different label but could be still related to the text query (for example, the image of the penguin is tagged as ”geography” but it is also correct under ”biology”). In figure 11 we have the top 5 images returned for the same queries using TextTopicNet. We can see how in the queries ”warfare” and ”biology” our model retrieved more relevant images than the ones retrieved by TextTopicNet. In the two queries about ”art” and ”geography”, they both have returned similar images (interestingly, in the ”art” query both models have retrieved images of buildings).

When we change the number of topics (table I) we can see that using 10 topics and 2 kernels improves the results to 0.34 in text to image retrieval and 0.32 in image to text. Changing the number of topics to 100 we generally obtain worse results, although they seem to improve when we use more kernels.

3) Classification: We can see the results on classification in table IV. TextTopicNet achieves a mAP of 0.48. Using the MDN model we obtain lower scores than TextTopicNet. We can also appreciate how the score decreases as we augment the number of kernels of the mixture, which happens with any number of topics. When the network is trained with 40 topics, the highest mAP obtained is 0.38 with one kernel while with 30 kernels the score drops to 0.32. We observe similar scores for all the mixtures with similar number of kernels, independently from the number of topics, which leads us to believe that a higher

Fig. 10: Top 5 images retrieved for 4 text queries belonging to different topics. The model uses a mixture of 3 kernels per image and we computed the likelihood of the text embedding for all the images. The number in parenthesis indicates the likelihood of every image.

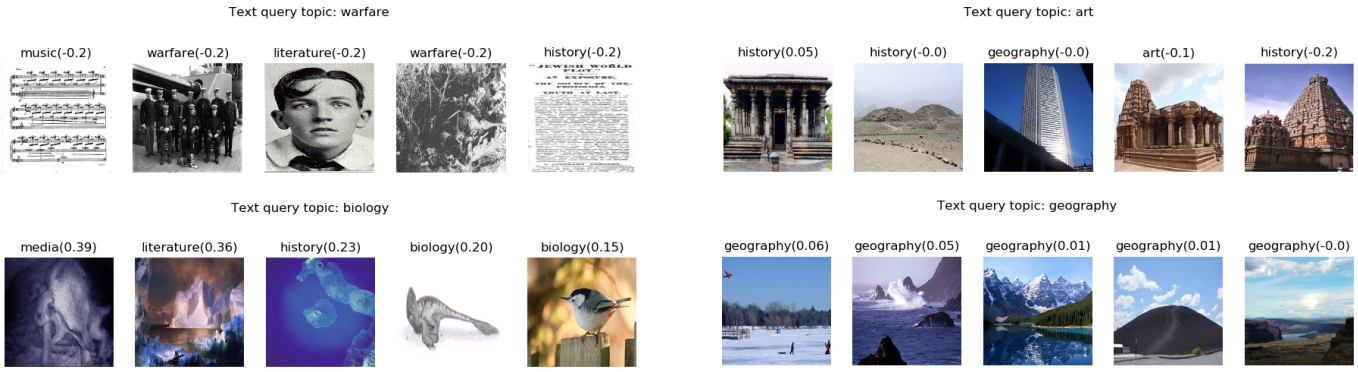


Fig. 11: Top 5 images retrieved for 4 text queries belonging to different topics using TextTopicNet. The number in parenthesis indicates the Kullback-Leibler divergence between the embedding of the image and the vectorial representation outputted by the network.

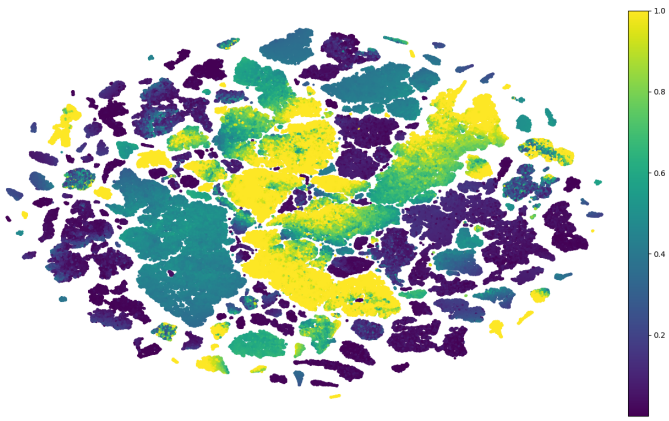
number of kernels seems to produce less discriminative features.

Having seen the results on the experiments we can say that the base framework (a MDN with Alexnet backbone, trained on the Wikipedia ImageCLEF dataset) manages to learn the image semantics but it does not perform better than TextTopicNet. On the proxy evaluation tasks it obtains worse results, and the visualization suggests that it does not fully exploit the descriptive power of the mixture models. We theorize that this may be caused by two reasons, and design two variations of the base framework to see if the results can be improved:

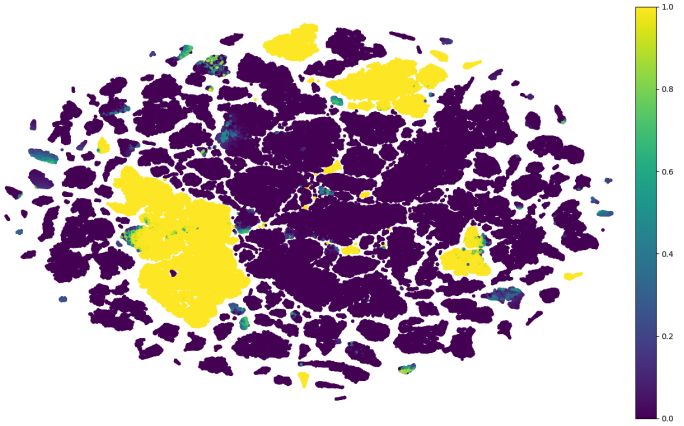
- Alexnet's descriptive power is too limited for the mixture model since we are using a more complex model to embed the images. In order to explore if using a more complex backbone would improve our results, we retrain both our architecture and the original TextTopicNet using ResNet-50 [14] as the CNN backbone.
- The dataset used is too noisy and small for properly training MDNs. Since every image corresponds to a whole article, we may have information in the textual embedding that is not entirely related with the image. We retrained both TextTopicNet and our MDN model using a larger dataset were the images and the text come also from the English Wikipedia, but instead of having pairs of articles and images, it comprises pairs of images and the section of the article they belong to. This way we expect the text articles to be more related with the images and hopefully generate a more clear topic space.

B. Using a more complex model (ResNet-50 + ImageCLEF Wikipedia dataset)

To see if our framework would benefit from having a more complex backbone CNN, we decided to train our network using ResNet-50 [14]. We also trained TextTopicNet with the same CNN to see how both networks improve with this change. We found that choosing the right parameters was harder than when we trained with Alexnet. When we trained our framework, we used Adam [19] with a learning rate of 6E-4, a learning rate decay of 0.1 every 20 epochs and a batch size of 86 images. In TextTopicNet we also used Adam with a learning rate of 5E-4, a batch size if 64 and a decay of 0.1 every 20 epochs. Our architecture was trained for 60 epochs while we trained TextTopicNet for 40 (in both cases, the models were trained until convergence).



(a) Sigma values when using 5 kernels and ResNet-50. The values have been clipped between 0 and 1 because there were some abnormally high values in some isolated kernels.



(b) Mixing coefficients when using 5 kernels and ResNet-50

Fig. 12: Variances and mixing coefficients when using 5 kernels jointly with ResNet-50. In both cases we have used random jitter to eliminate the possible overlap of the kernel means.

Fig. 13: Top 5 images retrieved for the same 4 textual queries from before using a mixture of 3 kernels, the number in parenthesis indicates the likelihood of every image.

1) Visualization: In figure 12 we can see the variance and the mixing coefficients when using 5 kernels and ResNet-50. In this case, the network manages to create multiple clusters with apparently low variance values. When we look at the mixing coefficients we can see that from all the kernels, there are a few of them that accumulate all the contribution to the mixture, while the majority do not contribute much. This is probably a similar phenomenon to the one observed in figures 7 and 8, where some of the kernels have a big contribution to the mixture while the others is low. This suggests that we are not taking proper advantage of the mixture of Gaussians approach, since only one of the kernels is contributing to the mixture.

2) Retrieval: All the results on retrieval using ResNet-50 are summarised in table II. TextTopicNet obtains almost the same results as in its Alexnet version, with a slight improvement in image to text retrieval and a slightly worse results in text to image. Our architecture obtains in general worse results with respect to its Alexnet version. In text to image we obtain the best results using 1 kernel and in image to text using 3 kernels. While we obtained considerably worse results in classification using 5 and 10 kernels (as we see in the next section), in retrieval they are similar to the results obtained with less kernels. We can see qualitative results using a mixture of 3 kernels in the figure 13.

3) Classification: Table V includes all the results of TextTopicNet and our architecture using ResNet-50 as the CNN. TextTopicNet improves its results up to a mAP of 0.55 (from the previous 0.48) when using ResNet-50. In our architecture the best scores are obtained using 1 and 3 kernels. With 1 kernel it goes up to 0.47 (0.38 using Alexnet) and 0.40 using 3 (from 0.36), but the results are still lower than in TextTopicNet. When using 5 and 10 kernels, the results are lower than in the base framework, and with 5 kernels we obtain a score of 0.26. This may be due to the lack of data to train more complex models such as the 5-kernel one.

C. Using a more complex dataset (Alexnet + new Wikipedia dataset)

To see if the results would improve using a larger and more fine-grained dataset we have trained TextTopicNet and our model using a new unreleased dataset [1]. This dataset contains more than 3 million images, of which approximately 2 million images are paired with their corresponding section of the article. The other 1 million images belong to the info box of the articles, and they are not directly related with any section. By having the section of the article every image belongs to, we expect that the semantic content of the text is more closely related to the image. Because of computing and time limitations, we used a subset of the provided dataset containing 600k pairs of sections and images. The LDA model trained used 40 topics.

Our network has been trained with a stochastic gradient descent, a learning rate of 0.025 with a momentum of 0.9 and a batch size of 64. The learning rate decay is set to 0.1 every 5 epochs. TextTopicNet has also been retrained on this dataset with stochastic gradient descent, a learning rate of 1E-4 with a momentum of 0.9 and a batch size of 64. The decay is also set to 0.1 every 5 epochs. In both cases they were trained for 12 epochs until convergence.

1) Retrieval: In retrieval the results obtained are similar as when training with the Wikipedia ImageCLEF dataset (table III. In this case we obtain the best results in text to image (0.32) using 10 kernels, while in image to text the best results are obtained using 1 kernel (0.25). Training TextTopicNet with this new dataset lowers its scores for text to image from 0.37 down to 0.35, and in image to text from 0.40 to 0.34.

2) Classification: In classification we managed to raise our scores up to 0.47 from 0.38 when using 1 kernel (table VI). Increasing the number of kernels yielded lower scores, as we have been seeing in previous results. TextTopicNet obtains almost the same scores as with the other dataset, obtaining a mAP of 0.47 (from a previous 0.48). In all the cases, the results on our architecture have improved when using 1, 5 and 10 kernels.

D. Results

In this section we collect all the results obtained with the previous experiments. In the tables I, II and III we show the results for retrieval using the base framework (I), using ResNet-50 as the CNN (table II) and using the new dataset (table III). For classification the results are presented in the same order in tables IV, V and VI.

In general, the time it took to train with Alexnet on the ImageCLEF Wikipedia dataset was almost the same in TextTopicNet and our model (about an hour of training for 60 epochs on a Nvidia GTX 1080 Ti). In both cases the training time using ResNet-50 increased to about 6 hours and was more memory intensive in the GPU. Training on the new Wikipedia dataset took around 7 hours.

Distance	Entropy	Entropy	Euclidean	Euclidean	Likelihood	Likelihood
Retrieval mode	Text to Image	Image to Text	Text to Image	Image to Text	Text to Image	Image to Text
TextTopicNet	**0.3763**	**0.4025**	-	-	-	-
MDN 40 Topics - 1 Kernel	0.1216	0.1199	0.3324	0.2611	0.3298	0.2607
MDN 40 Topics - 3 Kernels	0.1439	0.139	0.3339	0.2684	0.3064	0.2591
MDN 40 Topics - 10 Kernels	0.1456	0.1472	0.3327	0.2549	0.314	0.2555
MDN 40 Topics - 30 Kernels	0.1478	0.1534	0.3308	0.249	0.3113	0.25
MDN 10 Topics - 2 Kernels	0.3443	0.3268	**0.3443**	**0.3268**	**0.338**	**0.3066**
MDN 10 Topics - 3 Kernels	0.3274	0.3128	0.3328	0.2955	0.3264	0.296
MDN 10 Topics - 10 Kernels	0.3330	0.3127	0.3318	0.2899	0.3253	0.2901
MDN 100 Topics - 2 Kernels	0.1816	0.1662	0.3014	0.198	0.2912	0.1979
MDN 100 Topics - 3 Kernels	0.1478	0.1416	0.3067	0.2088	0.295	0.2088
MDN 100 Topics - 10 Kernels	0.1669	0.1519	0.3174	0.1927	0.3036	0.1927

TABLE I: Mean Average Precision scores obtained on the Wikipedia dataset [34] (image to text and text to image retrieval) using different distances and number of kernels.

Distance	Entropy	Entropy	Euclidean	Euclidean	Likelihood	Likelihood
Retrieval mode	Text to Image	Image to Text	Text to Image	Image to Text	Text to Image	Image to Text
TextTopicNet	**0.3704**	**0.4061**	-	-	-	-
MDN 1 Kernel	0.1717	0.1782	**0.3088**	0.2143	**0.3072**	0.2142
MDN 3 Kernels	0.1827	0.1900	0.2945	**0.2335**	0.3038	**0.2360**
MDN 5 Kernels	0.1438	0.1466	0.2926	0.2112	0.3014	0.2279
MDN 10 Kernels	0.1585	0.1526	0.2748	0.1723	0.2687	0.1824

TABLE II: Mean Average Precision scores obtained on the using ResNet-50 as the CNN and 40 topics.

[1]The dataset was provided by the research group, and it is not part of this project.

Distance	Entropy	Entropy	Euclidean	Euclidean	Likelihood	Likelihood
Retrieval mode	Text to Image	Image to Text	Text to Image	Image to Text	Text to Image	Image to Text
TextTopicNet	**0.3544**	**0.3497**	-	-	-	-
MDN 1 Kernel	0.1257	0.1186	0.3226	**0.2549**	**0.3192**	**0.2549**
MDN 5 Kernels	0.1598	0.1662	**0.3236**	0.2474	0.3177	0.2476
MDN 10 Kernels	0.1596	0.1656	0.3244	0.2486	0.3190	0.2488

TABLE III: Results on retrieval having trained with the new dataset.

Architecture	mAP
TextTopicNet	**0.48**
MDN 40 topics - 1 Kernel	0.38
MDN 40 topics - 3 Kernels	0.36
MDN 40 topics - 10 Kernels	0.35
MDN 40 topics - 30 Kernels	0.32
MDN 100 topics - 2 Kernels	0.36
MDN 100 topics - 3 Kernels	0.36
MDN 100 topics - 10 Kernels	0.35
MDN 100 topics - 30 Kernels	0.31
MDN 10 topics - 2 Kernels	0.36
MDN 10 topics - 3 Kernels	0.35
MDN 10 topics - 10 Kernels	0.35

TABLE IV: Results for classification on the Pascal VOC2007 dataset for TextTopicNet (first row) and our model for different number of kernels. We include the results for 10, 40 and 100 dimensions of the LDA space. The score of TextTopicNet is the one reported in [12], where they used 40 topics.

Architecture	mAP
TextTopicNet	**0.55**
MDN 1 Kernel	0.47
MDN 3 Kernels	0.40
MDN 5 Kernels	0.26
MDN 10 Kernels	0.32

TABLE V: Results for classification on the Pascal VOC2007 dataset using resnet as the CNN and 40 topics in the LDA.

Architecture	mAP
TextTopicNet	**0.47**
MDN 1 Kernel	0.43
MDN 5 Kernels	0.43
MDN 10 Kernels	0.41

TABLE VI: Results of classification having trained with the new dataset.

E. Analysis

Having seen the results we can say that the network, up to some point, manages to learn the semantics of the images and obtain acceptable results in retrieval and classification tasks. However, in all the cases, the results obtained using mixture density networks were still lower than the original TextTopicNet.

When we are using a low number of kernels, the figures from 7 and 9 suggest that only a portion of the images have a density function that clearly gives rise to different clusters in the embedding space. We can also appreciate how only one of the kernels usually has a high contribution to the mixture, while the rest of the kernels are less relevant. The semantics of these clusters (which included topics such as automobiles, planes or people) correspond to images that were not difficult to discriminate in TextTopicNet, while the rest of the images appeared to have overlapping kernels and were not separated in different topics. In either case, the density functions only had one predominant maximum. A higher number of kernels (as seen in figure 12, which was trained with ResNet) seems to give rise to different semantic clusters but once again only a portion of the kernels is relevant to the mixture while the rest have low contribution.

Increasing the number of kernels of the mixture generally lowers the performance of the network on the proxy evaluation tasks, especially in classification, where the results are always worse when we increase the number of kernels (particularly when we use ResNet, as seen in table V). In retrieval, the best results are usually obtained with less kernels as well, usually 1 or 2, although there are some exceptions like when the network was trained with 100 topics of LDA (table I). This can be explained with what we have seen with the visualization, that only one of the kernels of the mixture (or the overlap of several kernels) is relevant while the others just add more complexity to the model. The impact of the number of topics of the LDA space does not seem to be very substantial, although using 10 topics helped obtaining better scores in retrieval, but worse in classification.

We hypothesized that the reason behind these results was either the lack of descriptiveness of Alexnet or the article-level granularity of the dataset. Changing the CNN to ResNet-50 improved the results in classification, something that was expected, since this convolutional network is has a more deep and complex architecture. However, the results with TextTopicNet using ResNet were still better than the ones using MDNs. In retrieval the scores obtained were a bit lower than the ones obtained with Alexnet. With the new dataset the classification scores increased in our model, while in TextTopicNet they remained almost the same. This is possibly because of the higher number of training images (from 100k to 600 k), but as we have seen when we trained with ResNet, higher scores in classification do not translate to higher scores in retrieval. Having obtained lower scores than TextTopicNet in both cases, we assume that the network still did not take advantage of the MDN approach.

V. Conclusions

In this report we have presented a self-supervised learning method that embeds images into semantic spaces using Mixture Density Networks. In general, the numerical results obtained in classification and retrieval show that our network does not perform as good as the original TextTopicNet, the framework we based our architecture on. In fact, we obtain most of our best results using 1 kernel, which is basically reproducing TextTopicNet. Nevertheless, the visualization of the embedding space suggests that the network is learning better semantic representations, although we are not still taking full advantage of the descriptive power of the MDN since we only have one relevant maximum per mixture. While changing the model to a more descriptive one or using a larger, cleaner dataset did not seem to improve the results, they seemed to be the way forward to improve the results, although the initial steps we took during this work did not improve substantially our results.

References

[1] Pulkit Agrawal, Joao Carreira, and Jitendra Malik. Learning to see by moving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 37–45, 2015.

[2] Relja Arandjelovic and Andrew Zisserman. Look, listen and learn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 609–617, 2017.

[3] Christopher Bishop. Mixture density networks. Technical Report NCRG/94/004, January 1994.

[4] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[6] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430, 2015.

[7] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2051–2060, 2017.

[8] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[10] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*, pages 2121–2129, 2013.

[11] DN Geary. Mixture models: Inference and applications to clustering. *Journal of the Royal Statistical Society Series A*, 152(1):126–127, 1989.

[12] Lluis Gomez, Yash Patel, Marçal Rusiñol, Dimosthenis Karatzas, and CV Jawahar. Self-supervised learning of visual features through embedding images into text topic spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4230–4239, 2017.

[13] Albert Gordo and Diane Larlus. Beyond instance-level image retrieval: Leveraging captions to learn a global visual representation for semantic retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6589–6598, 2017.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] Olivier J Hénaff, Ali Razavi, Carl Doersch, SM Eslami, and Aaron van den Oord. Data-efficient image recognition with contrastive predictive coding. *arXiv preprint arXiv:1905.09272*, 2019.

[16] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[17] Dinesh Jayaraman and Kristen Grauman. Learning image representations tied to ego-motion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1413–1421, 2015.

[18] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[21] Thomas K Landauer, Peter W Foltz, and Darrell Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.

[22] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *European Conference on Computer Vision*, pages 577–593. Springer, 2016.

[23] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6874–6883, 2017.

[24] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196, 2014.

[25] Hsin-Ying Lee, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 667–676, 2017.

[26] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[27] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[28] Aravindh Mahendran, James Thewlis, and Andrea Vedaldi. Cross pixel optical-flow similarity for self-supervised learning. In *Asian Conference on Computer Vision*, pages 99–116. Springer, 2018.

[29] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[30] Andrew Owens, Jiajun Wu, Josh H McDermott, William T Freeman, and Antonio Torralba. Ambient sound provides supervision for visual learning. In *European conference on computer vision*, pages 801–816. Springer, 2016.

[31] Yash Patel, Lluis Gomez, Raul Gomez, Marçal Rusiñol, Dimosthenis Karatzas, and CV Jawahar. Texttopicnet-self-supervised learning of visual features through embedding images on semantic text spaces. *arXiv preprint arXiv:1807.02110*, 2018.

[32] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[33] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[34] Nikhil Rasiwasia, Jose Costa Pereira, Emanuele Coviello, Gabriel Doyle, Gert RG Lanckriet, Roger Levy, and Nuno Vasconcelos. A new approach to cross-modal multimedia retrieval. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 251–260. ACM, 2010.

[35] Amaia Salvador, Nicholas Hynes, Yusuf Aytar, Javier Marin, Ferda Ofli, Ingmar Weber, and Antonio Torralba. Learning cross-modal embeddings for cooking recipes and food images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3020–3028, 2017.

[36] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[37] Trieu H Trinh, Minh-Thang Luong, and Quoc V Le. Selfie: Self-supervised pretraining for image embedding. *arXiv preprint arXiv:1906.02940*, 2019.

[38] Theodora Tsikrika, Adrian Popescu, and Jana Kludas. Overview of the wikipedia image retrieval task at imageclef 2011. In *CLEF (Notebook Papers/Labs/Workshop)*, volume 4, page 5, 2011.

[39] Liwei Wang, Yin Li, and Svetlana Lazebnik. Learning deep structure-preserving image-text embeddings. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5005–5013, 2016.

[40] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2015.