Embedding Images and Sentences in a Common Space with a Recurrent Capsule Network

Danny Francis
Data Science Department
EURECOM
Biot, France
danny.francis@eurecom.fr

Benoit Huet
Data Science Department
EURECOM
Biot, France
benoit.huet@eurecom.fr

Bernard Merialdo
Data Science Department
EURECOM
Biot, France
bernard.merialdo@eurecom.fr

***Abstract*—Associating texts and images is an easy and intuitive task for a human being, but it raises some issues if we want that task to be accomplished by a computer. Among these issues, there is the problem of finding a common representation for images and sentences. Based on recent research about capsule networks, we define a novel model to tackle that issue. This model is trained and compared to other recent models on the Flickr8k database on Image Retrieval and Image Annotation (or Sentence Retrieval) tasks. We propose a new recurrent architecture inspired from capsule networks to replace the traditional LSTM/GRU and show how it leads to improved performances. Moreover, we show that the interest of our model goes beyond its performances and includes its intrinsic characteristics, which can explain why it performs particularly well on the Image Annotation task. In addition, we propose a routing procedure between capsules which is fully learned during the training of our model.**

***Index Terms*—multimodal embeddings, deep learning, multimedia retrieval**

I. Introduction

Humans can intuitively match sensorial perceptions such as vision with language. For instance, one can easily describe a landscape, or imagine how one looks like after reading its description in a novel. For a computer, that seemingly mundane task raises some questions: how to represent sentences and images and how to link these representations of two different modalities? Some successful works have been done in that field, such as automatic image captioning [1] or visual question answering [2].

Recently, big annotated databases of multimedia contents such as ImageNet [3] or Flickr8k [22] were built. These collections permitted the emergence of deep neural networks, because they can perform very well if they are trained with a sufficient amount of training examples. In computer vision, convolutional neural networks (CNN) [4], [5] are now widely used. In natural language processing, recurrent neural networks (RNN) such as long short-term memory units (LSTM) [6] or gated recurrent units (GRU) [7] are a common choice. These neural networks can learn representations of texts and images, and some works have shown that these representations can be used as a basis for comparisons [13], [16], [18]–[21]. One way of doing such comparisons is by defining a constrain on the neural networks that are used to represent the two different modalities, so that they learn the same representation for them: in that case they would output a vector representation of images or texts so that an image and a corresponding text are close according to some similarity measure. This is what our model is doing.

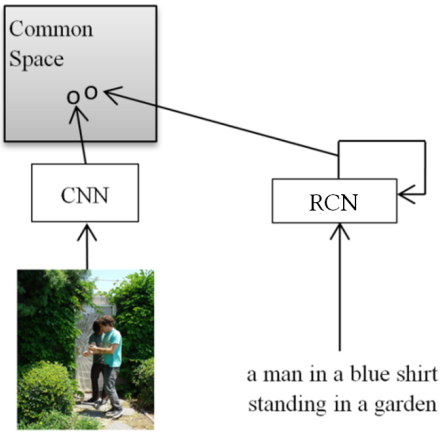

Fig. 1. Overview of our model: A first part computes image embeddings, a second one computes sentence embeddings through a Recurrent CapsNet. The two parts are constrained by a loss function to produce embeddings in the same multimodal space.

At the end of 2017, a novel architecture of neural networks called capsule [8] has shown promising results on some computer vision tasks. A capsule is a group of neurons whose role is to make some complex computations, and to output a simple vector as a result of these computations. These outputs are then routed towards higher-level capsules. During training they are supposed to specialize in the recognition of specific patterns; lower capsules learn to recognize simple shapes and higher capsules use the results of the computation of the lower capsules to recognize complex shapes. The advantage of capsules with respect to more common CNNs is that they do not contain pooling operations losing spatial information in high-level layers. We think that the idea of capsule-like units can be generalized to other fields than computer vision. The model we introduce in this paper uses capsule units to

analyze sentences we expect each capsule to perform well on a certain type of sentences. In our model, outputs of capsules are not routed towards higher level capsules but towards capsules of the same layer in a recurrent fashion: Fig. 1 depicts our proposed recurrent capsule network (Recurrent CapsNet or RCN) architecture. We will elaborate on this new deep model in subsequent parts of the paper.

Here, we present a novel model for visual sentence embeddings. It is divided into two parts: the first part computes sentence embeddings thanks to an RCN each capsule contains two GRUs as we will explain later on. The second part computes image embeddings based on feature vectors obtained with a ResNet-152 [5] (ResNet-152 often provides state-of-the-art results in computer vision tasks). We compare our model with some state of the art models used for image captioning and retrieval on the Flickr8k dataset [22] and show that our RCN performs better than most approaches and better than a simple GRU. The contribution of this paper is the definition of the RCN, and the presentation of its results on an Image Annotation task and on an Image Retrieval task when it is used to produce sentence embeddings.

The rest of the paper is organized as follows: in Section II we present some recent and related works. In Section III we give a formal definition of our model. In Section IV we present the results of experiments we made on our model. We conclude our paper in Section V.

II. Related Work

Several works have been done on building visual-semantic embeddings. In [23], Frome et al. made such embeddings through a model based on a skip-gram text modeling architecture [10] for the semantic part, and on AlexNet [4] for the visual part. Karpathy et al. proposed fragment embeddings in [12]: as the different parts of a descriptive sentence correspond to different parts of an image, they designed a model that would draw matches between image fragments and text fragments. The model proposed by Kiros et al. in [13] is similar to our model: they derive a sentence embedding with an LSTM and a features vector for a corresponding image through a CNN. Their objective function imposes a constrain on the two modalities so that they belong to the same space. The main difference between our two models is that we replace the LSTM by a novel Recurrent CapsNet architecture that is detailed in Section III.B. Variations of Kiros model have been suggested. Vendrov et al. have tried to induce an order between images and sentences by replacing the cosine similarity in the loss function by an order penalty [16]. More recently, Faghri et al. have shown that emphasizing hard negatives in the loss function would lead to high improvements [20].

Some other works are based on Fisher Vectors [23]. In [14], Klein et al. derive Fisher Vectors for sentences from a Gaussian Mixture Model (GMM) and a Hybrid Laplacian-Gaussian Mixture Model (HLGMM). In [15], Lev et al. propose an RNN that are used as generative probabilistic models. In these last two works, matches between images and texts are made through the Canonical Correlation Analysis algorithm [24]. In [17], Eisenschtat and Wolf also derive Fisher Vectors for sentences from a GMM and an HLGMM, but they do not use CCA to match images and sentences. They designed a 2-Way neural network that projects the two modalities onto a common space.

More complex architectures have also been designed in some papers. Niu et al. [18] have proposed a hierarchical multimodal LSTM: a sentence is parsed as a tree and correspondences between phrases and image regions are drawn. Nam et al. [19] proposed a Dual Attention Network that is designed to attend jointly to image regions and corresponding words in text. Gu et al. [21] proposed a model that would match texts and images by learning high-level global features, but also low-level local features thanks to two generative models: one of them generates sentences from images and the other one generates images from sentences.

The model we propose in this paper is inspired by a kind of neural network architecture called capsule [8], [9]. A capsule is a group of neurons that is supposed to do some complex and very specific computations, and then output a low dimensional vector as a result of these computations. In computer vision, this architecture is worthy of interest because it could overcome the problem of pooling operations in CNNs that lose all spatial information in higher layers. As far as we know, capsules have only been proposed in computer vision. Our model is the first one using capsules in natural language processing. Our architecture is also the first recurrent one: CapsNet have only been proposed in a feed-forward fashion. Moreover it has been shown in [27] that different language models did not perform equally on all sentences: some of them performed well on some sentences and some other on other sentences. This motivates the use of different specialized capsules to process sentences.

III. A Recurrent CapsNet for Visual Sentence Embedding

A. Gated Recurrent Units (GRU)

Gated Recurrent Units were introduced by Cho et al. in [7]. They are similar to LSTMs: they have similar performances and are well adapted to NLP because they can handle long-term dependencies in sentences. We preferred GRUs to LSTMs because they have less parameters for similar performances. More formally, a GRU is composed of an update gate z_s and a reset gate r_s, and can be described with the following expressions:

$$z_s = \sigma(W_{xz}x_s + W_{hz}h_{s-1} + b_z), \tag{1}$$

$$r_s = \sigma(W_{xr}x_s + W_{hr}h_{s-1} + b_r), \tag{2}$$

$$\tilde{h}_s = \tanh(W_{xh}x_s + W_{hh}(r_s \circ h_{s-1}) + b_h), \tag{3}$$

$$h_s = (1 - z_s) \circ h_{s-1} + z_s\tilde{h}_s, \tag{4}$$

with x_s the s-th input and h_s the s-th output or hidden state of the GRU. Here and throughout the paper, $\circ$ denotes the Hadamard product and σ denotes the sigmoid function.

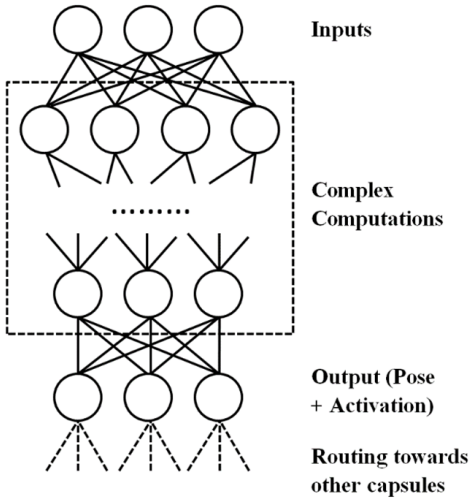

Fig. 2. A generic capsule for computer vision

B. Our Model

Our model can be divided into two parts: a first part aims at deriving sentence embeddings based on an RCN and the second part is designed to project an image features vector onto the same space as the aforementioned sentence embeddings. Let us first describe the first part of our model.

The idea behind capsules as they were designed by [8] for image processing is represented in Fig. 2. It consists in making complex computations and outputting a pose vector and an activation. This output is then routed towards subsequent capsules according to some predefined routing algorithm. The goal of that architecture is to have each capsule learning to recognize a visual feature based on what previous capsules have recognized before. For instance, some capsules could recognize eyes, a nose, a mouth and their respective positions. Then they would send their outputs to another capsule aiming at recognizing a whole face. It is architectured to avoid losing spatial information as common CNNs do, due to pooling operations. We think that capsules can also successfully perform other tasks such as NLP-related tasks, as our proposed model does.

In our case, each capsule contains two GRUs. The role of the first GRU is to output what we call a mask (the equivalent of the activation for computer vision) and the second one outputs a sentence embedding (the equivalent of the pose). The output of the first GRU of the i-th capsule is denoted by $\text{GRU}_{\text{mask}_i}(\text{input})$ and the output of the second GRU of the i-th capsule is denoted by $\text{GRU}_{\text{emb}_i}(\text{input})$. In the first GRU, the $\tanh$ function is replaced by a sigmoid function so that masks are composed of positive numbers. The mask that is output plays the role of an attention mechanism; we will give more details in the following. The biggest difference with capsules as they were described in [8] is that they are applied in a recurrent fashion: masks that are produced at time step t are applied to the input sentence, which is then fed into the same capsules at time step $t+1$. Sentence embeddings are built according to the following steps:

- we represent each word of a given sentence by a one-hot vector;
- we multiply each one-hot vector by a word embedding matrix;
- at each time step we apply masks onto the initial input sentence, and produce new masks based on the new input;
- we eventually output a sentence embedding.

Let us describe more formally how we compute that sentence embedding. Let s be a sentence. We encode each word of s with a one-hot vector: we have $s = (w_1, ..., w_L)$ with L the length of s, and $w_1, ..., w_L$ belonging to $\mathbb{R}^D$ with D the size of our vocabulary. Let $W_w \in \mathbb{R}^{D \times V}$ be the word embedding matrix. Then $x = W_w s$ will denote $(x_1, ..., x_L) = (W_w w_1, ..., W_w w_L)$ in the following. If m is a V-dimensional vector, then $m \circ x$ will denote $(m \circ x_1, ..., m \circ x_L)$. In what follows m denotes a mask and v denotes an embedding. N_c is the number of capsules. Embeddings are computed according to the following:

$$v_i^{(t)} = \text{GRU}_{\text{emb}_i}(m_i^{(t-1)} \circ x). \tag{5}$$

Masks are computed in two steps. First, capsules compute a mask according to the input sentence and the masks that were computed at the previous step:

$$\tilde{m}_i^{(t)} = \text{GRU}_{\text{mask}_i}(m_i^{(t-1)} \circ x). \tag{6}$$

Then, the m_i are computed as linear combinations of these masks, as follows:

$$m_i^{(t)} = \sum_{j=1}^{N_c} \alpha_{ij}^{(t)} \tilde{m}_i^{(t)} \tag{7}$$

and the final embeddings are computed in a similar way:

$$v^{(t)} = \sum_{i=1}^{N_c} \sum_{j=1}^{N_c} \beta_{ij}^{(t)} v_i^{(t)}. \tag{8}$$

The coefficients of the linear combinations are computed according to the following formulas ($\langle v_1 | v_2 \rangle$ denotes the scalar product between two vectors v_1 and v_2):

$$\alpha_{ij}^{(t)} = \frac{\left\langle v_i^{(t)} | v_j^{(t)} \right\rangle}{\sum_{k=1}^{N_c} \left\langle v_i^{(t)} | v_k^{(t)} \right\rangle}, \tag{9}$$

$$\beta_{ij}^{(t)} = \frac{\left\langle v_i^{(t)} | v_j^{(t)} \right\rangle}{\sum_{k=1}^{N_c} \sum_{l=1}^{N_c} \left\langle v_k^{(t)} | v_l^{(t)} \right\rangle}. \tag{10}$$

Please note that for $t = 0$, the masks are vectors whose coordinates are all equal to one: we actually just input sentences in the GRUs without applying any masks to them. These formulas can be interpreted in an intuitive way: if there are many capsules whose embeddings are similar to its own embedding, then it contributes a lot in the computation of

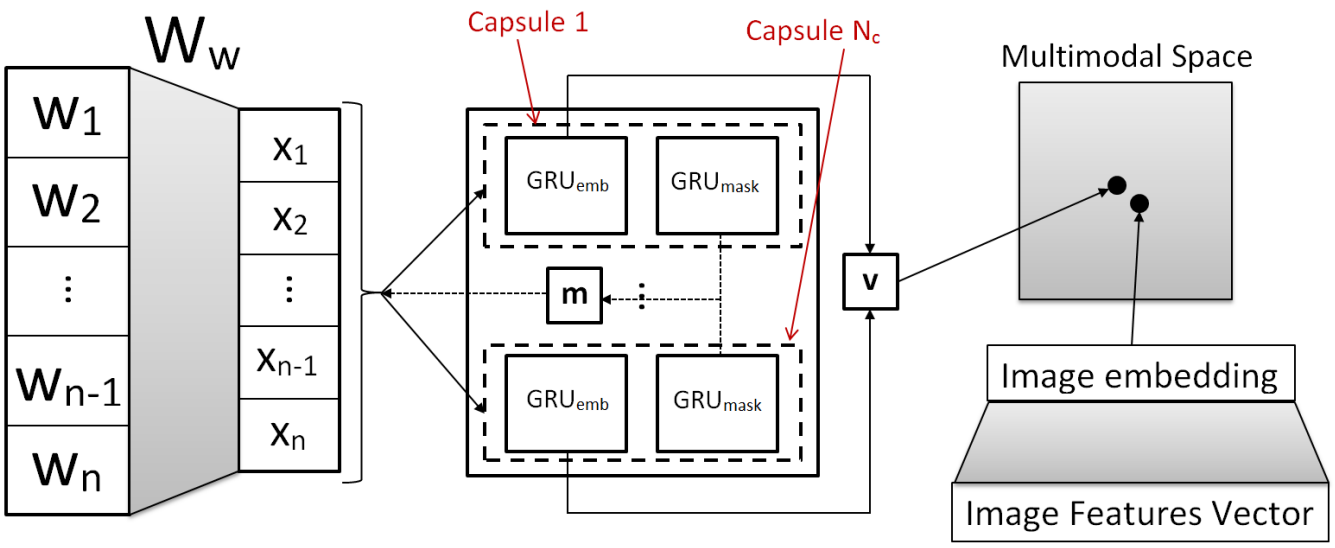

Fig. 3. Our model. Capsules are represented with dashed boxes. In the sentence embedding part, a sentence is represented by a sequence of one-hot vectors $(w_1, , w_n)$. It is transformed into a list of word embeddings $(x_1, , x_n)$ through a multiplication by a word embedding matrix W_w. The sentence then goes through the Recurrent CapsNet a pre-defined number of times, and eventually the RCN outputs a sentence embedding v. In the image embedding part, an affine transformation is applied to a features vector to obtain an image embedding. Both embeddings belong to the same multimodal space.

masks and embeddings. If its embedding is very different from other embeddings, the nits participation remains marginal. It can be viewed as a variation of the routing-by-agreement that is proposed in [8]: when other capsules output a sentence embedding that is very similar to the output of a given capsule, then this particular capsule plays an important role in the computation of the final embedding. If other capsules do not agree then it becomes more marginal. Regarding masks that are assigned to capsules, one can notice that a capsule contributes to the derivation of the mask of another capsule only if both capsules output a similar embedding. In that sense, embeddings that are output by capsules can be related to the pose vectors and masks can be viewed as activations as they were described in [8]: capsules send their activations to capsules agreeing to their pose. However, masks are not simple activations: multiplying term-by-term input words vectors by masks composed of positive numbers is more an attention mechanism that attends to a particular domain of the word embedding space.

The second part of our model is much simpler. First, we compute features vectors of images by keeping the output of the penultimate layer of a ResNet-152 that has been pre-trained on ImageNet 1000 classes and finetuned on MSCOCO [20]. More precisely we make nine crops of an image in the same way as it has been done in [16], and we compute the average of the corresponding features vectors of these crops. Then, we just apply an affine transform to these vectors. The parameters of that affine transform are derived during training. Fig. 3 gives a summary of how our model has been defined.

C. Loss Function

The loss function that we apply is the one that has been presented in [20]. If N_b is the number of samples in the mini-batch then we draw N_b image-sentence pairs $(u_k, v_k)_{1 \leqslant k \leqslant N_b}$ in the training dataset. Then, for each pair of the mini-batch we take the contrastive image and the contrastive sentence for which our model is less efficient, and apply a penalty for these contrastive image and sentence. More formally the loss function is defined as follows (ε is a hyperparameter):

$$L_1 = \frac{\sum_{k=1}^{N_b} \max_{l \neq k} \max(-\varepsilon, -\cos(u_k, v_k) + \cos(u_l, v_k))}{N_b} \quad (11)$$

$$L_2 = \frac{\sum_{k=1}^{N_b} \max_{l \neq k} \max(-\varepsilon, -\cos(u_k, v_k) + \cos(u_k, v_l))}{N_b} \quad (12)$$

$$L = L_1 + L_2 \quad (13)$$

L_1 is the ranking loss corresponding to the Image Retrieval task and L_2 is the ranking loss corresponding to the Image Annotation (or Sentence Retrieval) task. Optimizing the model boils down to finding parameters that minimize L.

D. Regularization

Since we want to avoid having the same masks in each capsule, and we do not want a capsule to output only zero-masks, we added the following regularization term to the loss function

$$V = \left(\frac{\sum_{i=1}^{N_c} \|m_i - \overline{m}\|_2^2}{N_c \|\overline{m}\|_2^2} \min_i \|m_i\|_2 \right)^{-1} \quad (14)$$

where $\overline{m}$ denotes the average mask. The new loss function is then

$$L' = L + \lambda V, \tag{15}$$

with λ a hyperparameter. As we will see in the next section, that regularization term can lead to better results.

IV. Results and Discussion

A. Parameters and implementation

We evaluated how our models performed on both the image annotation and the image retrieval tasks on the Flickr8k dataset [22]. This dataset comes with a predefined split between training, validation and testing samples; we use that split in our experiments. This dataset is composed of 8000 images with 5 sentences each: there are 6000 images in the training set, 1000 images in the validation set and 1000 images in the testing set.

Regarding the sentence embedding part of our model, we set its parameters as follows: we set the maximum sentence length to 16 (if longer the sentence is cut after the 16-th word), $D = 5000$ (we kept only the 5000 most common words and replaced all the other words by an UNK token) and $V = 300$. W_w was initialized according to precomputed word2vec embeddings [10]. Regarding the RCN, we found that a model with four capsules performed well. For a given sentence we kept $v^{(5)}$ as its corresponding embedding. For the image embedding part of our model, the dimension of image features vectors was 2048. The final embeddings dimension was 1024. Each GRU in our models has 1024-dimensional hidden states.

In the loss function, we set $\epsilon = 0.2$ and we tried different values for λ. We found that $\lambda = 0.05$ was giving good results. We trained our models using the Adam method [25] with mini-batches of 16 image-sentence pairs. The learning rate was 0.0002. We made all our implementations using the TensorFlow [26] library for Python.

B. Experiments

We compared our results on Flickr8k with some recent state-of-the-art models. In addition, we trained two different models on the Flickr8k dataset.

GRU. This model is simply the one we described in Section III.B, but with a simple GRU instead of the RCN. It is also a special case of our model where the number of capsules is 1 and the number of recursions is 0.

RCN-λ. This is the model we described in Section III.B. As mentioned in Section IV.A, we found that four capsules and four recursions lead to better results. λ is the same as in Section III.D.

As one can notice looking at Table I, results with our RCN are at state-of-the-art level. On top of that, capsules improve results of a single GRU, especially in the Image Annotation task. The regularization term seems to improve results for the Image Annotation task without having notable effects on the Image Retrieval task.

TABLE I
Results of our experiments on Flickr8k. R@K denotes Recall at rank K (higher is better). Best results among other models and among our models are in bold. Best results among all models are underlined

Model	Flickr8k					
	Image Annotation			Image Retrieval		
	R@1	**R@5**	**R@10**	**R@1**	**R@5**	**R@10**
Random	0.1	0.6	1.1	0.1	0.5	1.0
[14]	31.0	59.2	73.7	21.2	50.0	64.8
[15]	31.6	61.2	**74.3**	23.2	**53.3**	67.8
[18]	27	-	68.6	24.4	-	**68.1**
[17]	**43.4**	**63.2**	-	**29.3**	49.7	-
GRU	37.8	67.9	79.3	29.7	59.5	71.5
RCN-0	38.8	67.0	78.9	**30.3**	**60.4**	**72.5**
RCN-0.05	**41.5**	**70.6**	**81.2**	29.9	59.9	72.4

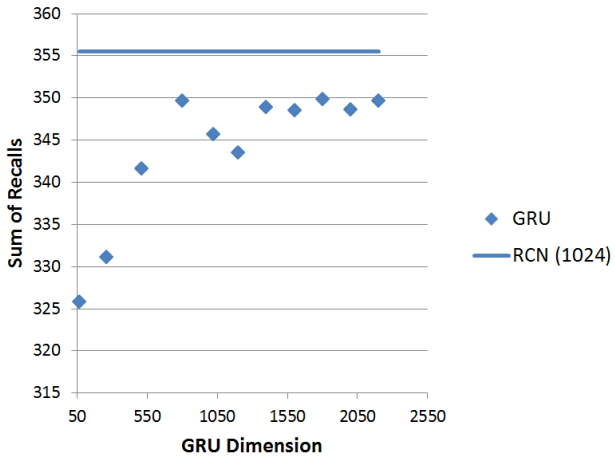

Fig. 4. Performance of GRUs on Flickr8k according to the size of their hidden states. The x-axis represents the size of hidden states, the y-axis represents the performance of the GRUs. This performance is the sum of R@1, R@5 and R@10 for both Image Annotation and Image Retrieval tasks. The line corresponds to the performance of our best model (RCN-0.05 with hidden states of dimension 1024).

C. Effect of the size of a simple GRU

In this section, we investigated if simply increasing the number of hidden states in a GRU could lead to the same improvements as our RCN. For that purpose, we computed the sum of the recalls at 1, 5 and 10 for the Image Annotation task and the Image Retrieval task for our best model (RCN-0.05) and for GRUs with 64 to 2200 hidden states.

As one can notice in Fig. 4, there is no improvement if the size of the GRU hidden states is increased above 800.

D. Effect of the regularization term

One can notice that the regularization term improves results with respect to simple GRU or non-regularized RCN. Why is that happening? Our hypothesis is that the regularization term imposes that capsules attend to different domains of the word embedding space. Therefore, sentence embeddings of different capsules tend to be more different than for a GRU or an RCN without normalization. As the Image Annotation task consists in retrieving one of five sentences for an image, the

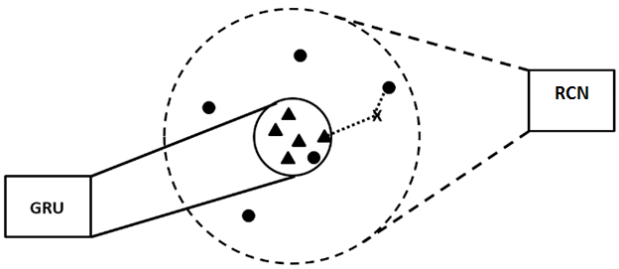

Fig. 5. Our hypothesis. The cross corresponds to an image, circles correspond to sentence embeddings output by an RCN and triangles correspond to sentence embeddings output by a GRU. In that case, both the RCN and the GRU output sentence embeddings for a given image around the same point, but even if the RCN generates worse embeddings on average, its best sentence embedding is better than the best sentence embedding output by the GRU.

distribution that capsules induce may lead to have one out of these five sentences being closer to its corresponding image. We summarized our hypothesis in Fig. 5.

V. Conclusion

In this paper, we proposed the RCN (Recurrent Capsule Network), a novel deep architecture for visual sentence embedding. It is based on the CapsNet architecture that was recently proposed in [8], but it differs from it in three important ways: we applied it to natural language processing, it is built in a recurrent fashion whereas the original CapsNet was built in a fully-connected fashion and the routing is performed using one of the GRUs.

We obtained some promising results, especially for the Image Annotation task where our RCN performs better than GRUs. We explained these performances improvements by the distribution that the capsules induced in the computation of sentence embeddings. In addition, we showed that the results of our capsules could not only be explained by the fact that they had more parameters than their corresponding GRUs: Fig. 4 showed that increasing the size of GRUs hidden states did not result in as good results as our models.

Some interesting directions of research would be to find how to take advantage of the distribution induced by our voting procedure for other tasks than sentence retrieval, or investigating how sentences are routed to different capsules: does it rely on semantics, syntax, or something else?

Acknowledgments

One of the Titan Xp used for this research was donated by the NVIDIA Corporation. This research is part of the GAFES project funded by ANR (the French National Research Agency).

References

[1] Karpathy, A., and Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3128-3137).

[2] Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Lawrence Zitnick, C., and Parikh, D. (2015). Vqa: Visual question answering. In Proceedings of the IEEE International Conference on Computer Vision (pp. 2425-2433).

[3] Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., and Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on (pp. 248-255). IEEE.

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

[5] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[6] Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[7] Cho, K., Van Merrinboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

[8] Sabour, S., Frosst, N., and Hinton, G. E. (2017). Dynamic routing between capsules. In Advances in Neural Information Processing Systems (pp. 3859-3869).

[9] Hinton, G. E., Krizhevsky, A., and Wang, S. D. (2011, June). Transforming auto-encoders. In International Conference on Artificial Neural Networks (pp. 44-51). Springer, Berlin, Heidelberg.

[10] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

[11] Frome, A., Corrado, G. S., Shlens, J., Bengio, S., Dean, J., and Mikolov, T. (2013). Devise: A deep visual-semantic embedding model. In Advances in neural information processing systems (pp. 2121-2129).

[12] Karpathy, A., Joulin, A., and Fei-Fei, L. F. (2014). Deep fragment embeddings for bidirectional image sentence mapping. In Advances in neural information processing systems (pp. 1889-1897).

[13] Kiros, R., Salakhutdinov, R., and Zemel, R. S. (2014). Unifying visual-semantic embeddings with multimodal neural language models. arXiv preprint arXiv:1411.2539.

[14] Klein, B., Lev, G., Sadeh, G., and Wolf, L. (2015). Associating neural word embeddings with deep image representations using fisher vectors. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 4437-4446).

[15] Lev, G., Sadeh, G., Klein, B., and Wolf, L. (2016, October). Rnn fisher vectors for action recognition and image annotation. In European Conference on Computer Vision (pp. 833-850). Springer, Cham.

[16] Vendrov, I., Kiros, R., Fidler, S., and Urtasun, R. (2015). Order-embeddings of images and language. arXiv preprint arXiv:1511.06361.

[17] Eisenschtat, A., and Wolf, L. (2017). Linking image and text with 2-way nets. arXiv preprint.

[18] Niu, Z., Zhou, M., Wang, L., Gao, X., and Hua, G. (2017, October). Hierarchical multimodal lstm for dense visual-semantic embedding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1881-1889).

[19] Nam, H., Ha, J. W., and Kim, J. (2016). Dual attention networks for multimodal reasoning and matching. arXiv preprint arXiv:1611.00471.

[20] Faghri, F., Fleet, D. J., Kiros, J. R., and Fidler, S. (2017). VSE++: Improving Visual-Semantic Embeddings with Hard Negatives.

[21] Gu, J., Cai, J., Joty, S., Niu, L., and Wang, G. (2017). Look, Imagine and Match: Improving Textual-Visual Cross-Modal Retrieval with Generative Models. arXiv preprint arXiv:1711.06420.

[22] Hodosh, M., Young, P., and Hockenmaier, J. (2013). Framing image description as a ranking task: Data, models and evaluation metrics. Journal of Artificial Intelligence Research, 47, 853-899.

[23] Perronnin, F., and Dance, C. (2007, June). Fisher kernels on visual vocabularies for image categorization. In Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on (pp. 1-8). IEEE.

[24] Hotelling, H. (1936). Relations between two sets of variates. Biometrika, 28(3/4), 321-377.

[25] Kingma, D. P., and Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[26] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... and Kudlur, M. (2016, November). TensorFlow: A System for Large-Scale Machine Learning. In OSDI (Vol. 16, pp. 265-283).

[27] Francis, D., Pidou, P., Merialdo, B., and Huet, B. (2017, April). Natural language access to video databases. In Multimedia Big Data (BigMM), 2017 IEEE Third International Conference on (pp. 78-81). IEEE.