Image Tag Refinement with Self Organizing Maps

Tolga Üstünkök
Department of Software Engineering
Atılım University
Ankara, Turkey
tolga.ustunkok@atilim.edu.tr

Ozan Can Acar
Department of Computer Engineering
Atılım University
Ankara, Turkey
ozan.acar@atilim.edu.tr

Murat Karakaya
Department of Computer Engineering
Atılım University
Ankara, Turkey
murat.karakaya@atilim.edu.tr

***Abstract*— Nowadays, data sharing has become faster than ever. This speed demands novel search methods. Most popular way of accessing the data is to search its tag. Therefore, creating tags, captions from an image is a research area that gains reputation rapidly. In this study, we aim to refine image captions by utilizing Self Organizing Maps. We extract image and caption pairs as feature vectors and then cluster those vectors. Vectors with similar content clustered close to each other. With the help of those clusters, we hope to get some relevant tags that do not exist in the original tags. We performed extensive experiments and presented our initial results. According to these results, the proposed model performs reasonably well with a 54% precision score. Finally, we conclude our work by providing a list of future work.**

Keywords— image tagging, tag refinement, self organizing maps, SOM, clustering.

I. INTRODUCTION

Tagging images with respect to patterns within images is an important research area. Increasing demand to smartphone market, results a digital image ocean. Large companies such as Facebook, Flickr need reliable image searching methods to locate the desired image in the digital image ocean. Thus, those images are needed to be categorized to make searching easy. In the past, tags were placed by humans. However, humans did not take the time to tag images properly. Hence, automatic image tagging methods are proposed [1-2]. Although they have decent image tagging success rates, they surely make mistakes.

Therefore, image tag refinement comes into play. Image tag refinement is the name of the process of removing imprecise tags and supplement incomplete tags [3]. Our aim in this paper is to enhance the existing tags by creating a new image tag refinement model. In this model, we adapted Self Organizing Maps (SOM) [4]- an unsupervised neural network model - to create clusters from combined image and tag feature vectors as an input to the refinement process.

The rest of the paper is organized as follows. Section II gives brief information about the methods that we have used. Section III defines our problem formulation as clear as possible. Section IV introduces our solution to the previously defined problem. Section V summarizes our findings and results. Finally, in Section VI, we give a summary of our method, problem and solution. We also suggest future works to improve our method.

II. BACKGROUND

We used VGG16 [5] as our image feature extractor network. VGG16 is a pretrained deep convolutional neural network architecture. It is trained on ImageNet: A Large-Scale Hierarchical Dataset [6]. VGG16 consists of 16 weight layers. The first 13 of those layers are convolutional layers and the rest is fully connected. For the first 4 convolutional layers, 2 max pooling layers are attached between every two layers. For the rest of the convolutional layers, 3 max pooling layers are attached between every three layers. After the convolutional section, 3 fully connected layers are present. The last of these layers is the softmax layer and it performs image classification. ReLU [7] is used as the activation function for all of the layers except for the last one. VGG16 can generalize other datasets as well. Thus, it is suitable for our purposes.

Sometimes tagging algorithms provide imprecise and incomplete tags for an image. At those times, a tag refinement process is needed. At its typical case, a tag refinement algorithm finds semantic-related images in training set for an image by using its tags. Then, it constructs a star graph from those candidate images based on visual similarity[3]. Finally, tags of those images are used to make necessary changes on the existing tag set to reach the final refined tag set.

Self Organizing Map (SOM) is an unsupervised neural network model which capable of clustering, regression, dimensionality reduction and data visualization. SOM tries to capture a discrete representation of the data being studied by exploiting the similarities in the input space. SOM has two layer; input and output layer. In the output layer, neurons are laid on a grid (generally 2 or 3 dimensional) in a way that they can represent the similar patterns in the input layer. Output neurons have neural weights inside them. Input layer has the training samples and generally training samples are the members of high dimensional spaces. Neural weights in the SOM must have the same dimension as the input data. With the help of their neighborhood and internal mechanics, neurons can project high dimensional data into a low dimensional one. By using these projections, we can visualize and cluster the input data. Unlike most of the neural network models, SOM utilizes a competitive - cooperative learning model. Learning process consist of a competition and cooperation parts. In the competition part all neurons compete against each other. Neuron with a weight which is most similar to the current input selected as the winner. Similarity metric is a predefined distance metric like Euclidean distance or cosine distance. Therefore winner neuron is the closest to the current train data. Henceforth, winner neuron and its neighborhood brought closer to the train data. In this part of the learning process, each winner neuron teaches its neighborhood about the training samples, hence it is called cooperative. Learning rate and the distance of the neighborhoods decay over time in order to prevent overfitting. There are various numbers of neighborhood configuration in the SOM literature [8].

III. PROBLEM DEFINITION

In this study, we employed Flicker8k dataset [9]. In this dataset, there are images with their corresponding captions. Each image has five captions. We aim generating additional new tags by utilizing the provided images and their captions. This process is called tag refinement.

978-1-7281-3992-0/19/$31.00 ©2019 IEEE

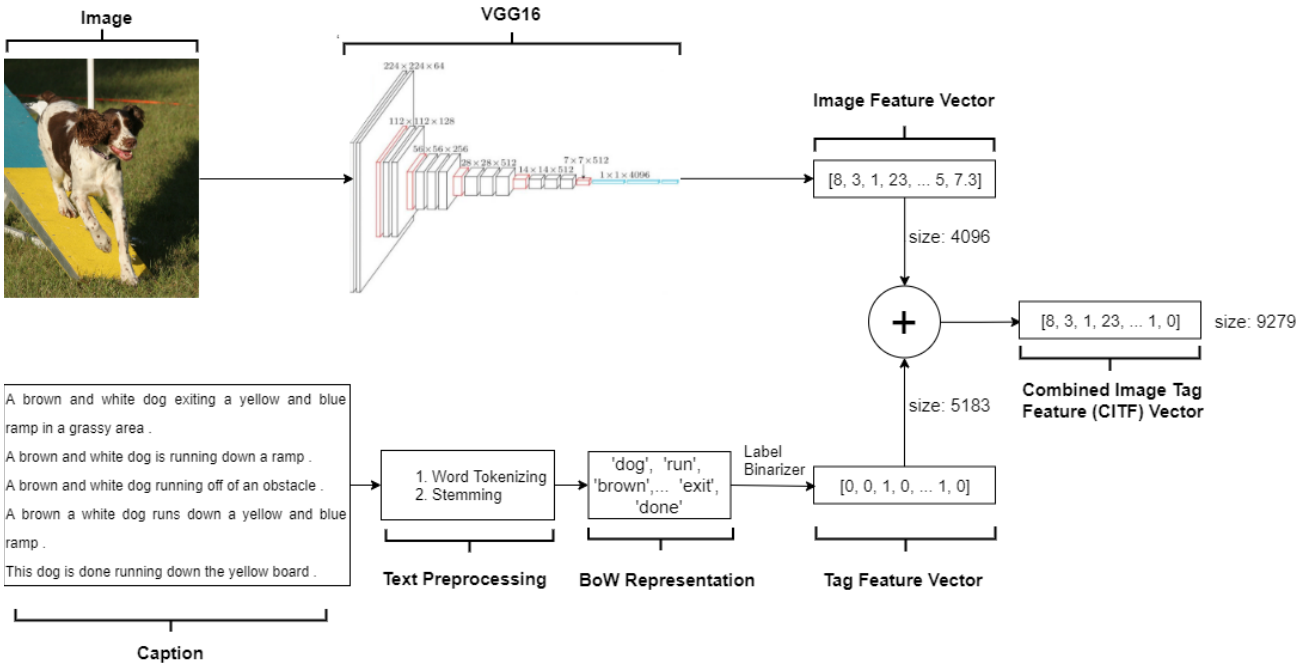

Fig. 1. The preprocessing pipeline and generation of *Combined Image-Tag Feature (CITF)* vectors. The upper part extracts the image features via VGG16 deep convolutional network model. The lower part creates bag of words (BoW) representations of given captions via a method called stemming. Then, creates a binary representation for the constructed BoW. Finally, both of those feature vectors are concatenated into each other.

There are extensive research about image tag refinement [10-12]. Most related works summarized as follows. Guillaumin et al. [10] employed a weighted nearest-neighborhood approach to predict image tags. Neighborhood weights are calculated according to neighborhood rank or distance. Sang et al. [11] proposed a method called Ranking based Multi-correlation Tensor Factorization (RMTF). This method tries to model the ternary relationship between user, image, and tag and reconstruct a more refined user-aware image tag associations. A ranking based model estimation is used to interpret the tagging data while estimating the model. In another study, a novel method called regularized Latent Dirichlet Allocation (rLDA) is used [12]. In this method, tag similarity and tag relevance are dealt simultaneously in an iterative manner. This situation allows to explore multi-wise relationships among tags.

Unlike other studies, we aim achieving tag refinement by exploiting the similarities in the latent subspace of similar images. We used SOM in order to get similar latent space representations together on image clusters. We also put previously added tags into the input vector of the SOM to hopefully get more logical clusters. By doing that, we expected to capture some relevant tags from similar neighbor images.

IV. Solution Method

Our solution method is composed of three steps. They are:

1. Preprocessing
2. Training
3. Evaluation

The details of these steps are explained in the following subsections.

A. Preprocessing Data

The data we've used for both testing and training purposes comes from the Flickr8k dataset. Flickr8k dataset has 8091 annotated images. To cluster the dataset with SOM, we preprocess it first.

The first stage of preprocessing is conducted on images. The process is shown in the upper part of the Fig. 1. We utilized VGG16 as feature extractor. Thus, we did not use the softmax output layer of the VGG16, but the latest fully connected layer. We passed each image from the VGG16 model and got the 4096 dimensional feature vectors as output.

The second one is converting annotations into Bag of Words (BoW). This process is shown in the lower part of the Fig. 1. In the dataset, each image has exactly five captions attached to it. We tokenized the captions, remove the stop words and perform a stemming algorithm to create BoW. Then, we encoded BoW into a binary representation. This allowed us to express BoWs with a one long string of 1s and 0s.

Then both image feature vectors and binarized BoW vectors concatenated to form the input vector of the SOM. This vector is shown as *Combined Image-Tag Feature (CITF)* vector in Fig. 1. Then we split them into training and test subsets. We used training subset in order to train a SOM model.

B. Training and Testing Operations

Originally Flickr8k dataset is partitioned into two subsets. The first part, training part, contains 6000 images and the second part, test part, contains 2000 images. Every image in Flickr8k has exactly five captions associated with them. We operated on the training subset. As discussed above, we applied preprocessing techniques on to these 6000 images. We obtained 6000 CITF vectors. Afterwards, we split 6000 CITF vectors into training and test subsets. We

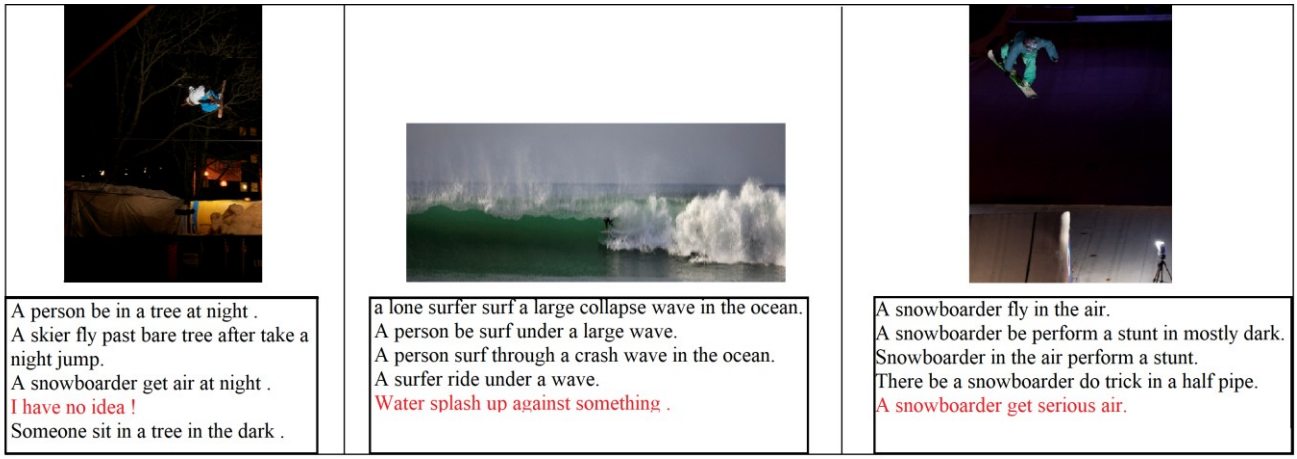

Fig. 2. The captions of the Flickr8k dataset is sometimes unreliable. Some captions (shown in red) are either irrelevant or nonsensical.

utilized 5900 of these vectors for training and 100 for testing. We passed the 5900 CITF vectors to the SOM as inputs to train it. Then, we tested the results with the remaining 100 unseen CITF vector.

C. Configuring the SOM Model

We made use of a package called MiniSom [13] to build our SOM model. We conducted extensive simulations for deciding on a proper hyperparameter set. First, we started with a relatively small grid size. Then, we discovered small grid sizes like 10x10 or even 30x30 cannot reflect the training data well enough. We decided a grid size of 80x80 would express the data, clusters and the neighborhood well. Due to the space limitations we only provided the hyperparameters that gives the best result. The decided hyperparameter values can be found in Table I.

D. Evaluation

After preprocessing stage, we trained our SOM model with 5900 CITF vectors and test its performance with 100 CITF vectors. We wished to employ the original captions as ground truth. However, when we analyzed the captions, we discovered some of them are not relevant with the corresponding image. An example is shown in Fig. 2. The captions marked with red are the unrealistic ones. Thus, we were not able to calculate a recall score for this dataset. Because of this drawback, we implemented two metrics for evaluation of the correctness of the generated tags. In the first metric, the generated tags are compared with the corresponding images by a human being, an observer. Observer marks whether a generated tag is correct or incorrect. According to these records, we calculated a precision score for each test image. Then, we took the average of all of the precision scores to get a final precision score. Precision scores is calculated according to the (1).

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

TP, true positive, means the tags created by the SOM that are relevant, meaningful with the test image. FP, false positive, means the tags created by the SOM are meaningless according to the test image.

In the second metric, we compared the SOM generated tags, T_G, with the original tags T_O for each given image. The difference operation was performed from SOM's point of view. You can see this operation in (2).

TABLE I. THE HYPERPARAMETER VALUES THAT WE USED IN OUR SOM MODEL.

Hyperparameter	Value
Grid Size	80x80
Neighborhood Function	Bubble / Circular
Radius (σ)	2
Learning Rate	0.5
Decaying Rate	Default
Epochs	2000
Distance Metric	Minkowski
Wight Initialization	Random Sampling

$$\text{Tags Refined} = T_R = T_G - (T_O \cap T_G) \quad (2)$$

T_G is the tag set that is generated by SOM. T_O is the original tag set that is obtained after the preprocessing of captions. The equation finds the intersection set of the original tags and proposed tags. Then, subtracts the intersection from the generated tags. Resultant set, T_R, is the set of tags that is generated by SOM which are not included in the original set T_O. So, these are the refinement proposals for the original tags. We named them as *Tags Refined (TR)*. Different tags between the two sets are also checked by the observer considering the corresponding image. Again, the observer marked the correct and incorrect tags and we calculated a precision score for each of the tags in T_R. Then, we took the average of the precision score.

As mentioned earlier, we randomly selected 100 samples from all of the CITF vectors for testing purposes. These 100 CITF vectors were not used while training SOM. After the training phase, we passed test samples to the SOM and found the winner neuron for each test sample. At each test sample, every winner neuron's and its circular neighborhood's weighted activation was calculated. This weighted activation forms the SOM's response to the current test sample. 60\% of the winner neuron's activation and 5% of every adjacent neighbors' activations summed in order to form the response. Relevant part of this activation was decoded to get the tags. Before decoding we only accept activation values above a certain threshold. With the help of some extensive experiments, we found this threshold as 0.3. After getting SOM's responses to each test image, we decoded them to find the SOM generated tags. Afterwards, we evaluated our model's precision according to the two metrics explained above.

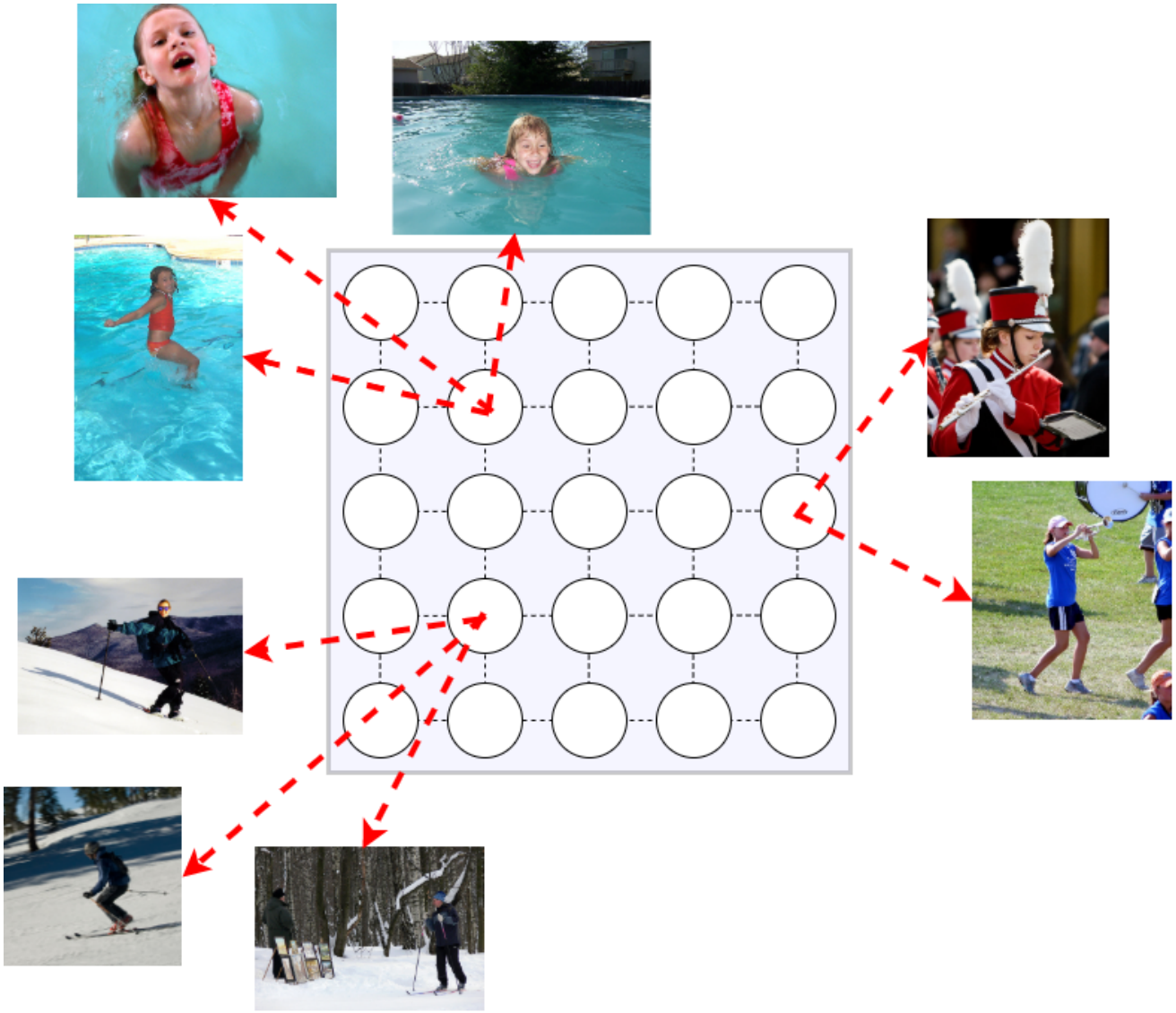

Fig. 3. With the help of the hit map, we can get an information about which CITF vector won the competition during training. We decode CITF vector's relevant part in order to visualize the image clusters. CITF vectors with similar content are clustered close to each other. Note that, those images are not our CITF vectors.

V. Results

We believe that clustering of CITF vectors were quite satisfactory. In order to provide some visual evidence, we extracted SOM's hit map. Hit map represents which neuron won for which training sample during the training phase. Then we decoded the relevant part of the CITF vectors in order to visualize the clusters. In Fig. 3, you can see the decoded images. As you can see, CITF vectors similar to each other clustered over the same neurons. Note that, we utilized CITF vectors during training. Fig. 3 only visualizes the results with images by using the relevant part of the CITF vectors.

We report our results in two categories. They are analytic results and visual results. Analytic results are the calculations that we performed to measure our methods accuracy. Visual results are the presentation of generated tags with its associated images.

A. Analytic Results

Our analytic results are based on observers' evaluation rather than assuming dataset captions as ground truths. As explained before, dataset contains incorrect tags and we cannot accept them as ground truths. The Table II summarizes our findings.

The average number of tags that are proposed by our SOM model is about 16. This means that for each of the newly seen images, our method proposes 16 tags in average without caring about if those tags exist in the original tag set (T_O). About 54% of those tags are completely relevant with the context of the corresponding image. Other 46% of the tags are either completely irrelevant or plausible in the sense of another similar context (i.e. grandstand balcony vs. deck balcony).

The average number of new tags that SOM proposes is about 13. This means that we propose approximately 13 new tags for a newly seen image. About 47% of those new tags are completely relevant with the image. The remaining tags are either completely irrelevant or relevant in an another similar context as above.

In addition to those results, we noticed that as the number of proposed tags increases, the accuracy decreases. We made

Fig. 4. Examples of the most accurate T_Rs. Black colored texts are the original tags and red colored texts are the generated tags different than the original tags (T_R). [Best seen in color]

Fig. 5. Examples of the least accurate T_Rs. Black colored texts are the original tags and red colored texts are the generated tags different than the original tags (T_R). [Best seen in color]

this observation by looking at the Pearson correlation coefficient between precision and number of tags for both sets. The coefficient is about -0.33 for both sets.

B. *Visual Results*

In this section, we present some images and their corresponding refined tags. The initial results are very promising.

In the Fig. 4, we put three of the most successful tag refinement examples. The black writings are the original tags that are given by the community and the red ones are the newly suggested tags. All suggested tags are comply with their corresponding images. Their precision is 100%.

Similarly, in Fig. 5 you can see the three worst tag refinements. All suggested tags are irrelevant with their corresponding images. Fortunately, we do not have many of them. Most of our results are between those two extremes.

VI. CONCLUSIONS

In this study, we aim refining given tags for an image. We are doing that by enhancing and enriching the already given tags. We utilized SOM to cluster the similar images and tags to borrow some relevant tags from different neighbor images. Best of our knowledge, there are no studies that utilizes SOM for the image tag refinement. Since this study is our initial work, we only present the preliminary results.

We observed some drawbacks in our problem. The weights of the image features and tags are not same for the SOM model. This is because the scale of the values does not match with each other. For example, the value of an image feature can float around 0 and 28, at the same time the value of the corresponding tag can only take values either 0 or 1. In addition, the distance metric of the SOM should be changed into a more suitable one (i.e. cosine similarity).

There are also some problems in Flickr8k dataset. The dataset is a heavily biased one. Almost all of the captions contain several repetitive words such as dog, man, woman, black, white, etc. Thus, in our future experiments, we plan to work on some other datasets.

To improve our SOM model, we plan to utilize more sophisticated SOM architectures (i.e. VQTAM [14]). We will also practice some changes in the weight update process of SOM. We will experiment with weight update methods that are more appropriate for image or caption attributes. We will also plan to merge the results of several randomly selected neighbor neurons to get possibly more improved results. We also believe that categorizing the image tags can increase our performance.

TABLE II. THE RESULTS OF OUR TESTS.

	T_G	T_R
Average Number of Proposed Tags	16.27	12.97
Pearson Correlation Coefficient	-0.35	-0.32
Average Number of Correct Tags	8.02	5.4
Average Number of Incorrect Tags	8.3	7.57
Average Precision	0.54	0.47

REFERENCES

[1] C. Qin, X. Bao, R. Roy Choudhury, and S. Nelakuditi, "Tagsense: a smartphone-based approach to automatic image tagging," in *Proceedings of the 9th international conference on Mobile systems, applications, and services*, 2011, pp. 1–14.

[2] M. Chen, A. Zheng, and K. Weinberger, "Fast image tagging," in *International conference on machine learning*, 2013, pp. 1274–1282.

[3] J. Fu and Y. Rui, "Advances in deep learning approaches for image tagging," *APSIPA Transactions on Signal and Information Processing*, vol. 6, 2017.

[4] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, 2009, pp. 248–255.

[7] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[8] C. A. Astudillo and B. J. Oommen, "Topology-oriented self-organizing maps: a survey," *Pattern analysis and applications*, vol. 17, no. 2, pp. 223–248, 2014.

[9] M. Hodosh, P. Young, and J. Hockenmaier, "Framing image description as a ranking task: Data, models and evaluation metrics," *Journal of Artificial Intelligence Research*, vol. 47, pp. 853–899, 2013.

[10] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid, "Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation," in *2009 IEEE 12th international conference on computer vision*, 2009, pp. 309–316.

[11] J. Sang, C. Xu, and J. Liu, "User-aware image tag refinement via ternary semantic analysis," *IEEE Transactions on Multimedia*, vol. 14, no. 3, pp. 883–895, 2012.

[12] J. Wang, J. Zhou, H. Xu, T. Mei, X.-S. Hua, and S. Li, "Image tag refinement by regularized latent Dirichlet allocation," *Computer Vision and Image Understanding*, vol. 124, pp. 61–70, 2014.

[13] JustGlowing, "MiniSom MiniSom is a minimalistic implementation of the Self Organizing Maps," 2019. [Online]. Available: https://github.com/JustGlowing/minisom. [Accessed: 03-Oct-2019].

[14] G. A. Barreto and A. F. Araujo, "Identification and control of dynamical systems using the self-organizing map," *IEEE Transactions on Neural Networks*, vol. 15, no. 5, pp. 1244–1259, 2004.