Natural Language Processing - Visual Question Answer

Juan Buhagiar – 11576014
Dimitris Nikolopoulos – 23456789
Sarantos Tzortzis – 11863331
https://github.com/sarantinio/Visual-Question-Answer--NLP.git
December 21th, 2017

Abstract

In the present paper, we introduce two models that aim to solve the Visual Questioning Answering (VQA) problem as accurately as possible. In VQA, we are given an image and a question about this image and we apply models which try to predict the most accurate and realistic answer. The first proposed model is a Continuous Bag of Words model and the latter a Long Short Term Memory model. Both of the models receive word tokens as an input which compose a question. Precomputed image features are concatenated to the output of these models in order to improve the classification process for a certain image. Our goal is to observe which model behaves better and what are the main differences between them. Different parameter setting were tested in each model and their results made it feasible to approach an answer to this goal. Overall, the LSTM model behaves better than the CBOW since it takes into account the long dependencies between words.

1 Introduction

Natural language processing is a research area that deals with the automatic interpretation of natural language, this could be in the form of text analysis, audio processing, text generation and many more. NLP is widely researched on many different languages and on many different tasks such as sentiment analysis (Pang et al., 2008), (Pak and Paroubek, 2010), named-entity recognition (Tjong Kim Sang and De Meulder, 2003), (Nadeau and Sekine, 2007) and many more.

Computer Vision is another broad research area which has become popular in recent years with the advancement in Convolutionial Neural Networks (CNNs). CNNs have been widely researched in recent years due to the advancements in processing capabilities of machines. Main topics which utilize CNNs are Object & Scene Classification (Krizhevsky et al., 2012), (Girshick et al., 2014), facial recognition (Turk and Pentland, 1991), (Wiskott et al., 1997), Medical Image analysis (McInerney and Terzopoulos, 1996) and many others.

There are many topics of research that involves these two big research areas together such as Image Captioning (Devlin et al., 2015), (Pan et al., 2004), content-based image retrieval (Gudivada and Raghavan, 1995) and Visual Question and Answering (Fukui et al., 2016), (Shih et al., 2016), (Xiong et al., 2016).

The problem we are addressing is Visual Question and Answering, in which we are given an image and a question related to the image and formed in natural language. The task is to produce a valid, accurate and expected answer formed in natural language. This task reflects the need of complex methods both from natural language processing and computer vision areas. Recent research has suggested that language can be used to obtain strong results without really understanding the visual content. This is ideal as we are focusing on the natural language processing aspect rather than computer vision.

This is a recent research area and related work has shown fair results, although there are many matters that arise. An example is the work performed upon the VQA dataset (Antol et al., 2015) which contains

thousands of images and at least three questions per image. Even though there are particularly nice results, it has been shown that the method contains biases toward certain answers given a question. For instance, if a question about sports is passed through a trained language model and the model outputs "tennis", then there is 60% chance that this is the correct answer. Another interesting observation done on the VQA dataset is that 87% of questions starting with "Do you see a.." have been annotated with the answer "yes" (Zhang et al., 2015).

In the present paper, we focus on building two language model approaches that try to solve the VQA problem as accurately as possible. The first model is a Continuous Bag of Words (CBOW) and the latter one is a Long Short Term Model (LSTM) upon word embeddings.

In many problems of Natural Language Processing, such as the present one, LSTM models perform generally better than CBOW models due to different reasons. Taking that into consideration, the research question of the present paper that we are posing concerns the training and the application of these two models upon the VQA problem. What are the differences between the performances of an LSTM and a CBOW model upon the VQA problem and which one behaves better? This question will try to be answered according to the given setting conditions and data, i.e.: a given calibrated VQA dataset and some pre-computed image features for each image from the dataset.

2 Problem

The task we are interested in for this research is the Visual Question and Answering (VQA). VQA involves answering natural language questions given an image in the most accurate way. This usually involves analyzing images and a question and producing the correct answer. A similar topic to this is image captioning which is focused on the generation of text from images that describes them. VQA is usually classified as a harder task to solve, as it involves finding answers not only from images but in combination with a question.

Very recent research has shown many approaches to solve the VQA problem, ranging from a customized neural networks to simple bag-of-words approach. Some research has suggested that simpler approaches have outperformed the state-of-the-art in this area (Zhou et al., 2015), (Kazemi and Elqursh, 2017). Therefore, in this research, we will focus on comparing a base approach with a more complex one to understand if the conclusions presented in this research are substantiated.

VQA has many interesting applications as it obtains a high level of understanding of images. It is one of the harder tasks of natural language processing and computer vision. If one was able to solve this problem to a reasonable degree then we would have a model that can look at an image, understand it's content, understand the question of the user and react. Applications of VQA range from better image retrieval to more complex artificial personal assistants. We will consider a bag-of-words approach as a simple method and an LSTM approach as a more complex method. These two ML approaches have been considered in previous research in the area and both have obtained good results. We will use already parsed image features for our research as the computer vision aspect is out of scope of our paper.

Visual question and answering (VQA) is a problem considered to be AI complete as it involves many of the fields of Artificial intelligence such as CV, NLP and KR. VQA involves a system that takes an image and a free-form natural language question about the input image and produces a natural-language answer as an output. Since we are dealing with open ended question the system must exhibit several AI concepts such as Object detection(eg. "What is the object on the table?"), activity recognition(eg. "What is the man doing?"), commonsense reasoning(eg. "Are those airplanes made out of metal?") and knowledge-base reasoning(eg. "Does the child have chicken pox?").

In most cases, question about images seek to obtain specific information and therefore would require very short answers. This makes it easier for us to evaluate the performance of any model produced by counting the number of questions the model predicts correctly.

We can observe from previous research that the easiest questions to answers are ones that ask for a yes or no answer and ones which require a number as a result. Other questions which require more complex logic are the ones that are the hardest for models

to generalize.

3 Approach

We have chosen to compare two different approaches to solve the VQA problem, both will use the same subset of a Question and Answers dataset. Our methods will also use the same image features obtained from the same dataset. We will first develop and test a simple continuous bag-of-words approach and compare it with a RNN approach. Both of which will be optimized by looking at various configurations and hyper-parameters.

3.1 Dataset

We are using a subset of the Visual Question Answering dataset (Antol et al., 2015) containing 60K pairs of questions and answers which has been balanced on different answer types. The image features that we will be using have already been extracted using the output layer of a pre-trained ResNet CNN which has been trained for image recognition.

Figure 1 illustrates some examples extracted from the dataset. These examples show that even though the answer is straight forward for humans it requires the machine to have a high level understanding of what the image represents.

The dataset contains 65 different question type, amongst the most popular is the 'how many' questions with approx 14.5K and the 'is the' questions with 4K. These question types n the dataset we also are usually asking for specific answers. We also have question types which are more complicated and less specific such as 'Why is the' and 'what is the name' which require greater knowledge reasoning about the image.

There are only three answer types: 'yes/no', 'number' and 'other'. The dataset is distributed equally across the answer types containing 20K of each answer. As the name indicated yes/no answers are binary answers that are either yes or no. Number answers are numeric answers and are typically associated with 'how many'-type of questions. Finally, other answers are more general and can include anything ranging from colours to random phrases such as 'above top of light post next to it'.

The answers for the questions where obtained by distributing the image and question to 10 individuals and each give an answers. Therefore, for each question and image we have 10 answers. The dataset also includes the most common answer from these 10. We will consider the most common answer as a class for our model.

To improve performance and reduce over-fitting, we are using a batch approach where we split the dataset into batches of variable size and process them separately. This also allows us to utilize more cores in our CPU resulting in less computation time.

3.2 BOW Model

The Bag-of-words (BOW) model is a very simple approach used in NLP and Computer Vision. BOW model uses the count of each word as a feature usually for some classifier. Despite its simplicity, it obtained fair results in many fields. Particularly in the VQA case, there has been research that shows the a simple BOW approach has similar results as more complex models.

The model we propose uses an extension of the bag of words approach called continuous bag of words (CBOW). CBOW tries to predict words given the context of a word. Typically, CBOW receives a vector of words (e.g. the words of the question) and creates a one-hot vector upon a certain vocabulary. These one-hot vectors are extremely sparse and the dimensionality decreases through a mapping to real-valued vectors. The goal behind this mapping is to map semantically similar words to similar real-valued vectors, called embeddings. Each word is mapped to an embedding of specific size and all the word embeddings are then summed or averaged and pass over a hidden layer which produces the output of the CBOW model.

In our proposed model, we use mini-batches of input questions to train it. Each mini-batch is passed as a set of one-hot vectors to a CBOW model which is defined by a hyper-parameter of its *embedding size*. The embeddings are summed and get concatenated with the image features.

The concatenated vector is passed through a classifier which has three hyper-parameters: the *number of hidden layers*, the *number of nodes* of each layer and the *transformation functions* applied in each layer. The output of the classifier is passed to a softmax layer which normalizes the results over a specific number of answer classes. The size of the

Figure 1: Examples from the VQA dataset (Antol et al., 2015)

last layer will be the same as the size of the answers found during training. We have tested different configurations of the hyper-parameters, using a variable size of the embedding matrix size, a different number of hidden layers and nodes and ReLu as the classifier's transformation functions.

3.3 RNN Model

Recurrent neural networks (RNN) have proven success in many different areas. They were recently introduced in the domain of natural language processing(Mikolov and Dean, 2013) and since then they have been applied to several subdomains. For example, RNNs have been very successful at sentiment analysis (Ghiassi et al., 2013) and machine translation (Cho et al., 2014). In recent research RNNs have been used for Visual Question and Answering and are the current state-of-the-art and they perform better than other language models. CBOW models perform well on giving a meaning to each word, however they are not able to relate the word to it context. On the other hand, RNN models are able to persist information in each stage.

In our problem, an RNN would use the information of one input, which in the present case is a word and pass it to the next word input. Intuitively, this would relate a word to its past context and therefore identify dependencies such as the form of a verb-answer which might depend on an word appearing in the question that defines the person of the verb. However, simple RNN models are not able to identify long-term dependencies of words in a question and answers. As the dependency gap gets wider, the harder it gets for a simple RNN model to predict a correct answer.

Long Short Term Memory network (LSTM) is able to overcome this disadvantage through its elegant design. An LSTM is a kind of RNN introduced by (Hochreiter and Schmidhuber, 1997) that is able to learn long term dependencies. For that reason, we are applying an LSTM model on the posed problem. An effective way of using RNN models is by appending it to the output of a n-gram model, such as the CBOW.

Our second model consists of a combination of the previously defined CBOW model and an LSTM model. More specifically, we add the LSTM model between the embeddings **E** produced by the CBOW model and the classifier The LSTM consists of a single layer which produces an output **c** defined as:

$$\mathbf{c} = LSTM(\mathbf{E}), \quad where \quad \mathbf{E} = \langle E_1, \ldots, E_i \rangle$$

E_i represents the embedding of the i-th word. Since we are using a mini-batch model, the input of the LSTM will be three-dimensional: **BxLxE**, where **B** is the batch size, **L** is the length of the longest sentence in the mini-batch and **E** is the embedding size. Output **c**, is the cell state produced by the last embedding, which is tuned according to the information passed from all the previous words of the question. This state is defined by its *hidden dimension* and it is the main hyper-parameter of our LSTM model.

The output matrix **c** is then concatenated with the visual information and the result is forwarded to the classifier. The classifier design remains the same as it was described in the CBOW subsection.

3.4 Loss & Evaluation Metric

The loss measure used is the Cross Entropy Loss. The loss is calculated at the classifier's last layer. This measure distributes the values of the final classes in such a way, so that the sum of all of the final values adds up to one. It combines the logarithmic Softmax with the negative log likelihood loss (NLL-Loss) and is given by the following formula:

$$loss(x, class) = -\log \frac{exp(x_{class})}{\sum_j exp(x_j)}$$

Finally, as an accuracy metric, we use the simple percentage of questions that were classified correctly. We also apply some simple techniques to avoid overfitting, by halting the training in case the validation set does not improve its accuracy for a certain amount of epochs.

4 Experiments & Empirical Evaluation

Our experiments were run on regular personal computers under the Linux environment. Our code was developed in Python, and our models were designed with the PyTorch module (v0.2.0). We ran individually the CBOW and then the LSTM model. Due to computational resource constraints, our parameter settings were set to the best maximum according to the capabilities of our computational resources.

4.1 Parameter Setting

Different parameter settings were tried out in order to investigate which one brings the best performances in our models. Each hyper-parameter was tested upon 100 and 500 epochs of training. The mini-batch size consisted by default of 128 questions. The learning rate was calibrated at 0.01 and is adjusted by the Adam optimizer. A final general parameter is the size of the pre-computed image features that is a vector of 2048 elements taken from each image.

First, we trained our CBOW model with the following configurations:

- Word embedding size: 64, 128 and 256.
- The classifier size (number of layers and nodes): 1 layer of 100 nodes / 2 layers of 100 and 300 (respectively) / three layers of 256 each. Each layer applies a ReLu transformation.

The setting of parameters in the CBOW that performed the best was kept the same for the LSTM model. The only hyper-parameter set is:

- LSTM hidden layer size: 128, 512 and 1024.

4.2 Results

Word embedding size	100 Epochs	500 Epochs
64	0.3065	0.3039
128	0.3244	**0.3269**
256	0.3224	0.3225

Table 1: Accuracy of the test set using Different Word Embedding Sizes

Classifier size	100 Epochs	500 Epochs
100	0.3065	**0.3084**
100 300	0.2964	0.3021
256 256 256	0.3059	0.3075

Table 2: Accuracy of the test set using different Classifier Size settings

LSTM state size	100 Epochs	500 Epochs
128	0.2906	0.3042
512	0.3130	0.3183
1024	0.3281	**0.3367**

Table 3: Accuracy of the test set using different LSTM State Size

	Overall	yes/no	number	others
CBOW	0.3269	0.5333	0.2540	0.1934
LSTM	0.3367	0.5431	0.2547	0.2103

Table 4: Result breakdown of answer types for the best performing parameters for CBOW and LSTM

Table 1 shows the accuracy of the test set on different word embedding sizes. The bold cells represent the best accuracy. We have trained our model on 100 and 500 epochs while using 64, 128 and 256 as word embedding sizes. The best performance was shown with a word embedding size of 128 during 500 epochs.

Table 2 shows the accuracy of the test set using different layer and node sizes for the classifier. Also, the embedding size was kept at 128, since it provides the best accuracy. The best performing classifier is the simple 1 layer with a size of 100 of 500 epochs.

Table 3 shows different tests performed with different LSTM layer sizes. We tested the LSTM with layer sizes of 128, 512, 1024. The 1024 layer size has the best performance during a 500 epochs training.

4.3 Analysis

From the results obtained, we can identify that the LSTM model works better then the CBOW on its own. We can also see that at 500 epochs the accuracy increases slightly. Unfortunately, we were not able to test higher epochs to find an approximate convergence.The classifier did not give better results which happens due to the small number of epochs. Apparently, the more hidden layer added, the longer training should be performed in terms of epochs. Therefore, we have not added a classifier layer and we simply kept a linear layer from the LSTM layer towards the classes.

To further understand our results we have re-run the models with the best performing parameters and obtained the performance for each answer type. We have three answer types 'yes/no', 'number' and 'others'. Table 4 summarizes the accuracy of the final two models upon those question types. In general, we can see that the 'yes/no' and 'number' answer types are equally generalized by both models. We see a small difference in the 'number' answer type between the two models. Overall, LSTM performed better than CBOW.

Figure 2 shows six examples from the VQA testset. Under each of the six images, we have included the top 3 predictions from both the CBOW and LSTM, depicted in green and blue colour, respectively. The correct answer are indicated with red colour.

Some observations can be made from these examples. Firstly, we can see that certain answers are not predicted correctly, however they are included in the top-3 accuracies. Another observation that can be made is that even though the model does not give the correct answer it manages to gives answers in the correct domain. For example, all prediction of 'What color is the dog?' include answers that are colours. There are instances that the model does not interpret fully the question and that is for instance, the question 'What is clear in the background?' for which we see answers that describe the object in the image but do not describe anything about the background.

Overall, it is pretty obvious that the LSTM model performed the best as expected because of several reasons. LSTM is able to identify dependencies upon the word embeddings of each word in a given question. These dependencies form the context of an answer and best the context is learned, the more accurate answers we receive. Apart from this, LSTM takes into account the ordering of the words of the input question and thus different order of the same words might produce different results. On the other hand, CBOW produces simple embedding that are trained to hold float values that represent the semantics of words. Moreover, the CBOW model does not take the word ordering into account. Thus, the resulted classes will be the same, no matter the ordering of the question.

5 Discussion and Conclusions

During this project, we approached the research question by applying the above experiments. According to the results, the LSTM model performs generally better by an approximate of 0.5% accu-

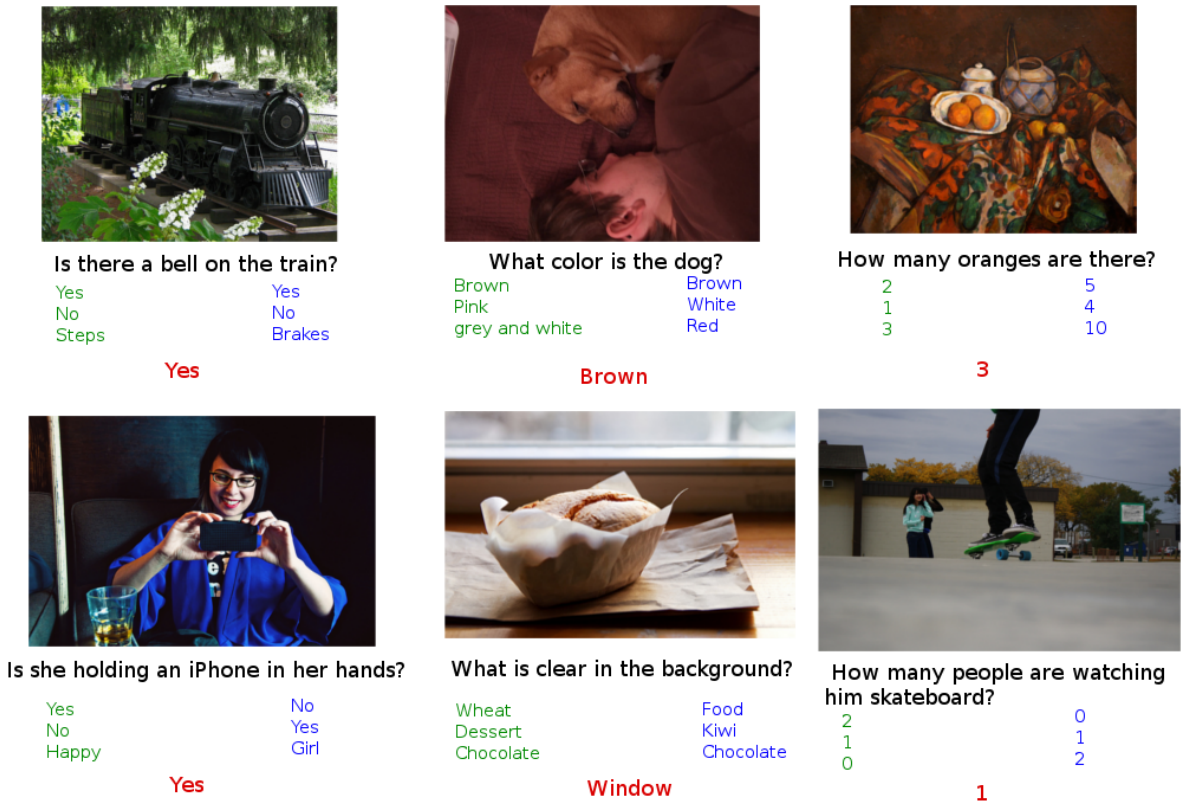

Figure 2: Predictions from our models on the test dataset. Black texts shows the question, Green text shows the CBOW top3 predictions, Blue text shows the LSTM top3 predictions and the red text shows the correct answer.

racy in comparison to the CBOW model. LSTM is able to make better predictions, because of the way it is designed which makes it able to identify long dependencies between words in a question. CBOW gives fair results but it does not outperform LSTM.

Several papers have presented related research and presented best results. For instance, the previously mentioned paper from (Zhou et al., 2015) presented similar CBOW, LSTM models that achieved accuracies of about 55%. Moreover, they investigate Convolutional Neural Network models that were applied upon the image features. This work was the baseline for our proposed models.

Nonetheless, we believe that certain limitations did not let the model improve. Such limitations were for example the limited dataset and the computational resources with which we run the training. We expect that the present project will be improved in order to achieve better results by overcoming these limitations.

In the future, it is desirable to investigate how the models react to higher epochs of training and where they converge. For this stronger machines would be good to be found in order to perform the training faster. Furthermore, we believe that we can obtain better results by increasing the sizes of the datasets, as seen in other research in this area. Moreover, other datasets for Visual Question Answering would be interesting to be tested in order to find how the models performs upon them. Finally, other image features should be investigated and not only the precomputed ones we obtained.

All in all, the topic of the Visual Question Answering is still recent and more research is expected to be applied. It will be particularly compelling to identify a state-of-art model that solves this problem in the most efficient and accurate way. A model that will be able to solve open-ended answers in a very

realistic way.

6 Team responsibilities

Throughout this project, we met at least once a week to discuss the current state of the project and adjust out plans. We were all working individually on coding, by splitting tasks. Dimitris was mostly responsible for the LSTM model and the layer-to-layer transitions, Juan worked on the input from the dataset and the CBOW model and Sarantos worked upon the dataset split, termination condition and the evaluation metrics. The code was pushed on a GitHub repository, to which we all had access and made us able to inspect each other's commits. The distribution of the work went well and was even between us, both in the report and the coding parts. The report was generated through a common sharelatex document to which we all contributed. The overall approach allowed for a smooth and efficiently collaborative process in the project.

References

[Antol et al.2015] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2425–2433.

[Cho et al.2014] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

[Devlin et al.2015] Jacob Devlin, Hao Cheng, Hao Fang, Saurabh Gupta, Li Deng, Xiaodong He, Geoffrey Zweig, and Margaret Mitchell. 2015. Language models for image captioning: The quirks and what works. *arXiv preprint arXiv:1505.01809*.

[Fukui et al.2016] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. 2016. Multimodal compact bilinear pooling for visual question answering and visual grounding. *CoRR*, abs/1606.01847.

[Ghiassi et al.2013] Manoochehr Ghiassi, James Skinner, and David Zimbra. 2013. Twitter brand sentiment analysis: A hybrid system using n-gram analysis and dynamic artificial neural network. *Expert Systems with applications*, 40(16):6266–6282.

[Girshick et al.2014] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587.

[Gudivada and Raghavan1995] Venkat N Gudivada and Vijay V Raghavan. 1995. Content based image retrieval systems. *Computer*, 28(9):18–22.

[Hochreiter and Schmidhuber1997] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

[Kazemi and Elqursh2017] Vahid Kazemi and Ali Elqursh. 2017. Show, ask, attend, and answer: A strong baseline for visual question answering. *arXiv preprint arXiv:1704.03162*.

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

[McInerney and Terzopoulos1996] Tim McInerney and Demetri Terzopoulos. 1996. Deformable models in medical image analysis: a survey. *Medical image analysis*, 1(2):91–108.

[Mikolov and Dean2013] Sutskever I. Chen K. Corrado G.S. Mikolov, T. and J. Dean. 2013. Distributed representations of words and phrases and their compositionality. In *In Advances in neural information processing systems*, pages 3111–3119.

[Nadeau and Sekine2007] David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

[Pak and Paroubek2010] Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10.

[Pan et al.2004] Jia-Yu Pan, Hyung-Jeong Yang, Pinar Duygulu, and Christos Faloutsos. 2004. Automatic image captioning. In *Multimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference on*, volume 3, pages 1987–1990. IEEE.

[Pang et al.2008] Bo Pang, Lillian Lee, et al. 2008. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135.

[Shih et al.2016] Kevin J. Shih, Saurabh Singh, and Derek Hoiem. 2016. Where to look: Focus regions for visual question answering. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June.

[Tjong Kim Sang and De Meulder2003] Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the*

seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4, pages 142–147. Association for Computational Linguistics.

[Turk and Pentland1991] Matthew A Turk and Alex P Pentland. 1991. Face recognition using eigenfaces. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*, pages 586–591. IEEE.

[Wiskott et al.1997] Laurenz Wiskott, Norbert Krüger, N Kuiger, and Christoph Von Der Malsburg. 1997. Face recognition by elastic bunch graph matching. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):775–779.

[Xiong et al.2016] Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. In *International Conference on Machine Learning*, pages 2397–2406.

[Zhang et al.2015] Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2015. Yin and yang: Balancing and answering binary visual questions. *CoRR*, abs/1511.05099.

[Zhou et al.2015] Bolei Zhou, Yuandong Tian, Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. 2015. Simple baseline for visual question answering. *arXiv preprint arXiv:1512.02167*.