Dialogue Based Image Retrieval Project

Joe Harrison
11770430

Romeo Goosens
10424458

Jorn Ranzijn
11138610

Abstract

In this project two approaches, classification and regression, of text-based image retrieval were tested on the MSCOCO dataset. The aim is to pick the correct image from a line-up of 10 images given a dialogue of 10 lines and a caption about the target image. The dialogue and caption were modified using various pre-processing techniques for all models except the baseline model. The obtained text data of each sample was transformed into a word embedding by either summing up the embeddings of each word or using a long short-term memory network. This resulted in three types of models that only differed in the way their word embeddings were created. In each of the classification models, word embeddings were concatenated with the image features and used as input for a multilayer perceptron. The image with the highest probability was predicted to most likely be the target image. The second approach, regression, only used the dialogue and caption as input and tried to recreate the target as output. The model that pre-processed the text data and used the standard word embedding outperformed the other models on 3 out of 4 benchmarks. After qualitative analysis, however, we found that the LSTM model, despite not having the best quantitative scores, does have a better understanding of high-level linguistic concepts.

1 Introduction

Imagine overhearing a conversation between two people talking about an image, after which you are asked to pick the image that these people were talking about from a line-up of 10 images. You would almost instantly pick the correct image. This is something that comes natural to humans and even young children are capable of successfully performing this task. AI systems, however, have tremendous difficulties with this endeavour because they don't 'understand' what the two people are talking about. Interpreting language is a large research area within natural language processing and formally it is called Natural Language Understanding. Natural Language Understanding is considered to be an AI complete problem which serves as an indicator of how hard the problem really is and how rewarding it could be if a solution is found (Popescu et al., 2003). During the project we were interested in using natural language for image retrieval. The problem we were trying to solve is as follows: given some text that belongs to an image, select that corresponding image from a line-up of 10 different images. Ideally, this would mean that from the text our model would be able to infer what the text is about and use that information to decide what image is most accurately described by that text. Image retrieval has received a lot of attention and with the steady increase of image data on the web this will probably continue to be the case. Two different strategies were applied to solve the problem. The first combined image features with text features and predicts a score for every combination. The second was a sequence to sequence method in which the text features were used to predict an image feature vector.

1.1 Background

The problem we are dealing with intersects with fields such as Visual Question Answering (VQA), Visual Dialogue and Text-based image retrieval. In Visual Question Answering (VQA) the task at hand is answering a question pertaining to an image. This is fairly similar to visual dialogue in which a conversation is held about an image but where also the history of the conversation is taken into account. In text-based image retrieval, the goal is to find an image based on some textual features. In our project, the dialogue about the image, a caption describing an image and the image itself

are provided. This means that we no longer have to predict any textual data but our model should still be able to extract meaning from the image and text in ways that are similar to techniques used in the previously mentioned fields. In VQA tasks, there have been attempts to answer the question using only the textual data (caption and dialogue) (Goyal et al., 2017). This already achieved fairly good results but the combination of visual and textual features can improve performance even more (Zhou et al., 2015). (Zhou et al., 2015) suggest concatenating the image features and word embedding and feeding these combined features through a softmax function. In our problem the output values are different as they are now referring to an image instead of possible answers to questions.

1.2 Problem statement

In this project we aim to retrieve the correct image from line-up of 10 images given a caption of the target image and a dialogue of 10 lines about the target image. In this paper we compare different techniques to achieve this. We built three classification models, of which we set one as a baseline for model comparison. Besides the classification models we also built one regression model. We are expecting the LSTM model to perform well as this takes into account the ordering of the words. We assume that the dialogue and captions in the dataset are clear enough for humans to pick the correct image from the line-up.

2 Method

In this section the dataset and methods used to build the classifiers are discussed.

2.1 Dataset

We used Microsoft's 2014 version of the Common Objects in Context (MSCOCO) dataset (Lin et al., 2014). This dataset has been used in a variety of classification tasks such as the ones described in the background section. For this specific project a dataset has been created in which each sample contains a set of 10 image feature vectors, a dialogue of 10 lines concerning the target image and a caption of the target image. There are two difficulties: easy and hard. In the easy dataset the 10 images in each sample are chosen at random. In most cases the caption and dialogue only concern the target image. In the hard dataset 4 images

(a) Easy example

(b) Hard example

share the same object as the target image. For example: the target image contains a dog, and 4 images also have dogs in them 1b.

Both the easy and hard dataset were split into 40000 training samples, 5000 validation samples and a test set of 5000 samples. The test set was only used for quantitative and qualitative analysis.

2.2 Pre-processing

All models, except for the baseline model, use preprocessing. For each sample the 10 lines of dialogue and the caption are put together and converted to lowercase. We then remove all punctuation and words that are in a predefined set of stopwords from the Natural Language Toolkit (NLTK). Within the NLTK, we used the Snowball stemmer instead of the Porter stemmer because it resulted in a higher accuracy for all models. Pre-processing lead to a substantial reduction of unique words in our training set. Before preprocessing the training dataset contained 21997 unique words excluding special tokens for unknown words (<UNK >) and padding (<PAD >), while the pre-processed dataset contained 15456 distinct words. Each word in the training dataset was given a unique index so that sentences could be represented as lists of integers. Words in the test or validation set that did not appear in the vocabulary would get the index of the aforementioned <UNK>token. We experimented with the removal of additional words that were most uncommon and most common but saw no significant improvement.

2.3 Model description

Two different approaches, classification and sequence to sequence, were used in order to solve the problem of selecting the image that most likely corresponds to a piece of text. Each model was trained for 30 epochs.

2.3.1 Classification

The first type of model was based on a classification approach in which the image features and the text embeddings were combined in order to produce a score for every text-image combination.

This classification approach resulted in three types of models. The first was a simple baseline model, as shown in figure 2. This baseline functioned as a model to which the performance of the other models could be compared.

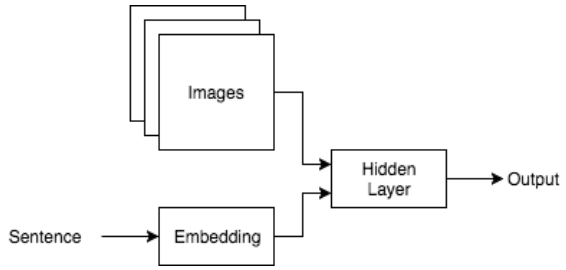

Figure 2: Overview of the baseline model

In the baseline model, every word was converted to a word embedding of size 50 using the Pytorch embedding function (Paszke et al., 2017). (Lai et al., 2015) found that the performance of word embeddings does not increase significantly with its dimensionality after a certain size for specific NLP tasks. Testing the performance of word embedding for 50, 100 and 200 dimensions resulted in similar performance, so we opted for the smaller embedding to reduce overall training time. The embedding for every word in the text was summed and then concatenated with an image vector resulting in a 2098-dimensional vector. This vector was used as the input for a multilayer perceptron (MLP). The MLP had an input layer with 2098 input nodes, a single hidden layer with 45 nodes and a single output node. The size for the hidden layer was determined with the help of the geometric rule proposed by Masters (Masters, 1993), see equation 1.

$$\#\text{hidden neurons} = \sqrt{\#\text{input neurons} \times \#\text{output neurons}} \quad (1)$$

For the hidden layer a sigmoid activation function was used and the output was kept linear as the cross-entropy loss function automatically applies a softmax. The model was optimised using the Adam optimization algorithm (Kingma and Ba, 2014). The second classification model was very similar to the baseline. The only difference was the use of pre-processed text instead of raw text, as shown in figure 3. For the specifics of pre-processing see section 2.2. We will refer to this model as the CBOW model.

The third classification model used pre-processing and a long-short-term memory model

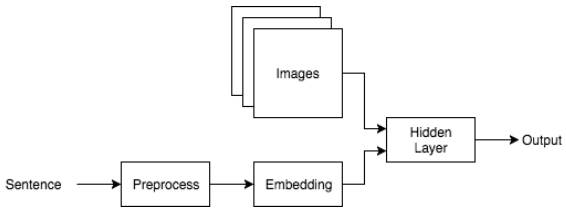

Figure 3: Overview of the classification model with pre-processing

(LSTM) to create a word embedding. The difference between the previous word embedding is that the word embedding of the LSTM was no longer a summation of the embeddings of the individual words. Instead, the words are sequentially fed into the LSTM which then returns a single word embedding, containing information from all the individual words that appeared in the text. The LSTM takes previous word embeddings into account while creating its output embedding. This new embedding was also of size 50 and concatenated with the image features. The classification was done in the same fashion as the other classification models, using an MLP with the same architecture and a cross-entropy loss function, see figure 4.

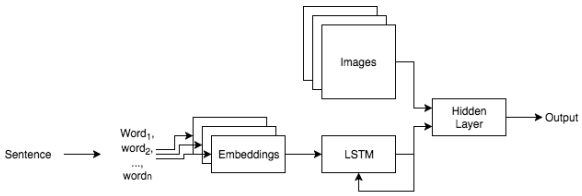

Figure 4: Overview of the classification model with pre-processing and LSTM

2.3.2 Sequence to Sequence

The second type of model was based on a sequence to sequence approach. This model no longer uses any image features but tries to predict the right image based solely on the text features. The sequence to sequence model used the pre-processed text and word embeddings were created using the Pytorch embedding function, see figure 5.

The word embedding had a dimension of 50 which was used as the input for the regression model. The MLP used for this model had 50 input nodes, a hidden layer with 320 nodes and an output layer of 2048. The hidden layer used a sigmoid activation function while the output layer

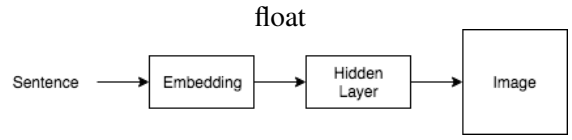

Figure 5: Overview of the classification model with pre-processing and sequence to sequence approach

was linear. The model was trained using the mean squared error loss function and the Adam optimization algorithm. The MLP transformed the text features into image features, after which a similarity metric was used to decide which of the 10 images was most similar to the predicted image. The cosine similarity metric was used for this task, see equation 2.

$$similarity = \frac{\sum_{i=1}^{N} A_i B_i}{\sqrt{\sum_{i=1}^{N} A_i^2}\sqrt{\sum_{i=1}^{N} B_i^2}} \quad (2)$$

Here A and B are the image feature vectors of dimension N, with $N = 2048$.

2.4 Learning behaviour

We needed to choose an appropriate learning rate for each model. In order to do this we chose a learning rate for each model for which its loss would generally go down smoothly with each iteration. All models eventually had a plot similar to figure 6. Some models, such as in figure **??**, experience abrupt rises in loss after approximately 60 iterations. This could indicate that the learning rate could be set even lower, or that the loss is unable to decrease after a certain number of iterations.

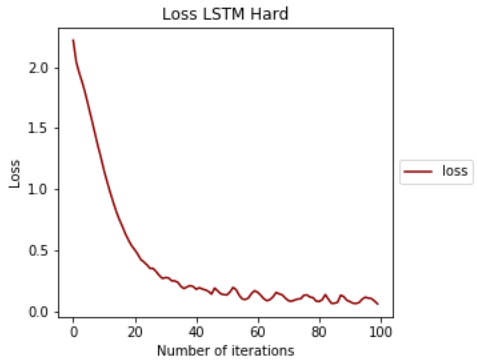

Figure 6: Example loss per iteration plot. Model used: LSTM

Model	Difficulty	Top 1	Top 5
Baseline	Easy	0.8978	0.9954
Baseline	Hard	0.4126	0.9142
Preprocessing	Easy	0.9166	**0.996**
Preprocessing	Hard	**0.4324**	**0.9152**
LSTM	Easy	0.8064	0.9814
LSTM	Hard	0.285	0.8216
Seq2Seq	Easy	**0.9304**	0.995
Seq2Seq	Hard	0.3878	0.895

Table 1: Results per model on test set

3 Quantitative analysis

The top one and top five accuracy were calculated for every model on the hard and easy datasets. Top one accuracy indicates how many times the model picked the correct image from the line-up of 10 images. The top five accuracy is a good indication to see if a model performance better than random guessing. It calculates how many times the target image appeared in the top five images that had the highest probability out of the 10 images.

3.1 Model performance

The classification model that had pre-processed data as input outperformed the baseline and all other models on the top one hard, top five easy and top five hard benchmark. The sequence to sequence model had the highest score on the top one easy benchmark. Combining text and image in the input gives a slight advantage over just using text as input in the top one. For an overview of the achieved performance scores for every model, see table 1.

3.2 Accuracy plots

For each model the validation accuracy and training accuracy were plotted against the number of iterations, as shown in figure 9 in the appendix. In all models trained on the easy dataset the top five accuracy of both the training data and validation data quickly converge near perfect scores. In the baseline model and preprocessed CBOW model we see that these models show signs of severe overfitting. The difference between the accuracy on the training set and on the validation set is very large in these models. For the models trained on the hard dataset every model, except the sequence to sequence model, suffers from overfitting in both the top one benchmark and top five benchmark.

4 Qualitative analysis

For each model we extracted samples from the test set on which the model performed very well and very poorly. These images were used in the qualitative analysis where we try to get some insight into what images the models have trouble with and what images are easier to predict. The obtained images were visualized in order to show the models prediction ranking. The highest ranked image is placed on the left and the lowest on the right. The target image is outlined in green.

Figure 7a, 7b, 7c, 7d show four different top 10 classifications of the same sample. The baseline model is unable to correctly classify the image in the top one, however, the first six images all contain zebras. When reading the dialogue it is unclear why the baseline ordered these images in this way. The sequence to sequence model, CBOW classifier and LSTM classifier all have a correct top one classification and it is relatively reasonable why these images have this ordering. It looks like they have a better interpretation of concepts like 'grass' and 'trees'. The first four predictions of the LSTM model contain zebras eating grass, and the remaining images of zebras not eating grass come right after these four predictions. This might indicate that the LSTM model has a better understanding of the concept of eating grass. When looking at the sequence to sequence model, it is noticeable that the pigeons are grouped together which is not the case for the other models.
It looks like the concepts *zebra* and *trees* are better understood than concepts such as *further* or quantities such as *three* or *two*. Therefore, it seems that every classifier has a better understanding of lexical semantics compared to compositional semantics.

Figure 8a shows an incorrect prediction of the CBOW classifier on a sample from the hard dataset. The main things that become apparent from this sample is that although the target is incorrect, the classifier is still able to group all the images according to the animal they contain. This might mean that the concept of animals is clearly understood and it even has degrees of similarity between other animals.

Another important issue is that images with a lot of negations in their answers are harder to predict correctly. When looking at the dialogue of figure 7a there are only two negations in the answer, namely "*Are there clouds?cant see because its too bright*" and "*Is there water in the picture? no*". When looking at figure 8a and 8b there are a lot more negation answers which results in a bad accuracy. The reason for this might be that each model is just using the word embeddings and these embeddings have no information about the context itself. The dialogue of figure contains a lot of answers that repeat the question and then has answers in the negated form, i.e "Are there any trees? *there are no trees*" and "Can you see the sun? *I cannot see the sun*". In these examples the embedding of the word trees and sun is given twice to the classifier. When analyzing the ordered images it is noticeable that the first two images contain both trees and suns while the dialogue said that there weren't. Also, the dialogue and captions are about a skating boy but when looking at the correct image (the last one) it is very hard to tell that there actually is a skate board. This could be the reason why our model puts the target image in the last place and images where a skate board is easily distinguished upfront.
To get a better understanding of the concepts that are learned during the training phase a query based approach is used that returns the an ordered set of images based on a typed query. Given ten images and a query, the model orders the images based on the query and by chancing the query a different order of images is returned. When testing concepts of the 10 images about wildlife (Figure 7a, 7b, 7c, 7d) it is noticeable that concepts as "Zebra" or "giraffe" are well understood by the model. It appears that the models have good intuitions about objects and there surroundings. However when using concepts like "*further*" and "*near*" or numerical information like "*two*" and "*three*" it is clear that these concepts are not fully learned by the model. Therefore, dialogues with information about the number of entities or any word that is a compositional semantic is not adding more information for a better classification.

5 Discussion

The baseline model that used the raw data performed worse than the model that used preprocessing. This indicates that there is noise in the data which the baseline tries to model, leading to

(a) Baseline classifier

(b) CBOW classifier

(c) LSTM classifier

(d) SEQ2SEQ regressor

Figure 7: Three classifications models the uppermost is , with the dialog: "Is the setting appear to be in the wild ? *yes* Is there water in the picture ? *no* How many zebras are there ? *5* Are they close up or far away ? *3 are close and 2 are further* How many gazelle are there ? *3* Are there trees ? *yes, a lot in the distance* Can you see the sky ? *yes, it's very bright white like a light* Are there clouds ? *can't see because it's too bright* Is the grass tall or short ? *short* Are the zebras eating the grass ? *yes*" , and the caption: "A herd of zebras and gazelle grazing in a green field". Two answers in a negation form.

a worse performance. It came as a surprise that the LSTM model performed worse than the two other classification models as it includes sequential information which is important in sentence semantics. Similar behaviour was observed in (Zhou et al., 2015), although they showed that increasing the amount of training data causes the LSTM model to outperform other models. Therefore, a similar strategy might be worthwhile to pursue in this project in order to improve the LSTM model's performance. The model that used preprocessing clearly performed best on the hard dataset while the sequence to sequence model performed best on the easy dataset. The performance of the two different types of models therefore really seems to depend on the type of dataset used. It is possible that sequence to sequence models works better on data where the target values clearly differ from the other images, and classification models can still work well on target values that are closer to each other. Further research would be necessary to determine if that is really the case. An option would be to create multiple datasets, each having a different amount of images that belong to the same category, and see how well both types of models perform. Based on the research done in this project, one would expect the classification model with preprocessing to outperform sequence to sequence when the number of pictures from the same category in a sample increases.

In the future we would also like to take into account more text features which is based on our qualitative analysis. If for example the dialogue contains the question 'does this scene contain a penguin?' and the answer is 'no i don't see a penguin'. Then it is likely that an image containing a penguin will falsely be predicted to be in the top one. Taking negations into account might result in a better performance. From the qualitative analysis it also became clear that certain concepts are better understood by the model than others. In the future, it would be useful to test specific concepts and retrain the model on the concepts when they are not fully understood. This will results in better understanding of the information given by the caption and/or the dialog. Also, it would be better to test certain concepts on all the images rather than 10 per sample.

It would also be interesting to see whether the outputted image feature vector of the sequence to se-

(a) Is this in the wild or a zoo? *i'm not sure but i think this is in captivity somewhere* Can you see any buildings? *no it is open field but does not look like africa* Are there any other animals visible? *no* Are there any trees nearby? *not just grass* Is there anything else remarkable about this photo other than the zebra?*no i'm sorry to say it is just a zebra in a grass fieled nothing else at all* Is the zebra looking well groomed? *yes it looks healthy and clean* Is the zebra running or walking? *walking* Is the zebra white with black stripes or black with white stripes? *white with black stripes* Can you see the sky? *no the image is too focused* Is this a boy zebra or a girl zebra? *it is definitely a girl*, and the caption: "A zebra walking through a green grassy area". Five answers in a negation form.

(b) " How old is the boy? *i don't know* Are there any cars? no what color is his hair? *hair is covered* Is he wearing shorts? *no* Is he wearing a helmet? *no helmet worn* Are there other skaters around? *no* How's the weather? *the weather is cold* Can you see any houses? *I see no houses* Can you see the sun? *I cannot see the sun* Are there any trees? *there are no trees*" , and the caption: "A boy in a green jacket skateboarding on a street" Seven answers in a negation form.

quence model could be transformed back into an image. We could then observe whether this image resembles in any kind what the caption and dialogue are about. Performing qualitative analysis on these constructed images could help find ambiguities in the text, and help engineer additional features.

6 Conclusion

We set out to find the best performing algorithm for the task of picking the correct image out of a line-up of 10 images given a caption and 10 lines of dialogue concerning the target image. It appears that applying pre-processing techniques to the input data is important as it improves model performance. We expected the LSTM model to perform well because of the inclusion of sequential information. However, this was not observed as it performed even worse than the baseline. Performance of the LSTM model might increase with more data. We also tried recreating the image feature vector from just the input text data in the sequence to sequence model. This model performed the best on the easy dataset but worse on the hard dataset. Therefore, it is hard to draw any strict conclusion as to which model is better quantitatively. It is possible that a sequence to sequence model works better for tasks in which target values are widely distributed and classification works better in the opposite case. However, further research is necessary to draw this conclusion. We conclude after qualitative analysis that the LSTM model, despite having the worst performance quantitatively, seemed to grasp high-level concepts the best out of every model.

7 Team responsibilities

Jorn mainly focused on the effects of different types of pre-processing techniques and the function of embeddings. Joe mostly focused on the implementation of the models and hyper parameter tuning. Romeo mostly focused on qualitative analysis, creating plots and Cuda parallelisation. The writing of the report was evenly divided. Everyone gave their input on every task necessary for this project.

References

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Siwei Lai, Kang Liu, Liheng Xu, and Jun Zhao. 2015. How to generate a good word embedding? *CoRR* abs/1507.05523. http://arxiv.org/abs/1507.05523.

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

Timothy Masters. 1993. *Practical neural network recipes in C++*. Morgan Kaufmann.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zem-

ing Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch .

Ana-Maria Popescu, Oren Etzioni, and Henry Kautz. 2003. Towards a theory of natural language interfaces to databases. In *Proceedings of the 8th International Conference on Intelligent User Interfaces*. ACM, New York, NY, USA, IUI '03, pages 149–157. https://doi.org/10.1145/604045.604070.

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

8 Appendix

8.1 Accuracy plots

Accuracy or learning curve plots for the baseline, cbow and LSTM models

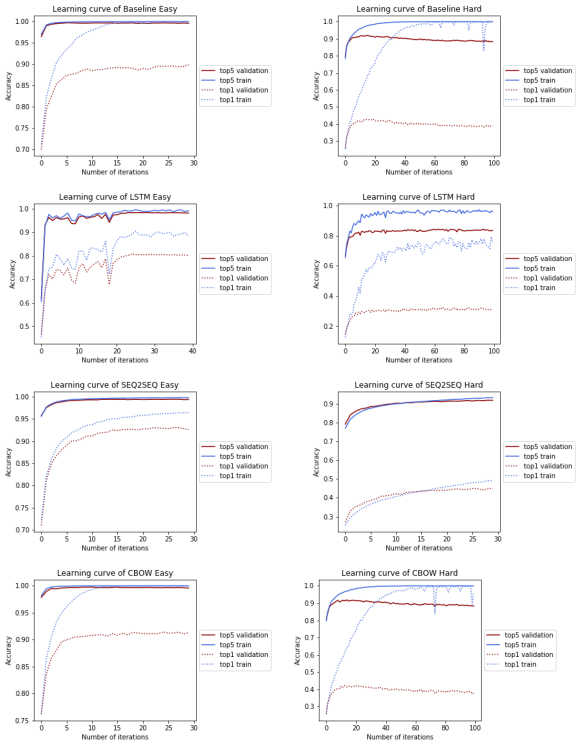

Figure 9: Accuracy or learning curve plots for four different models

8.2 Loss plots

Loss plots for the baseline, cbow and LSTM models

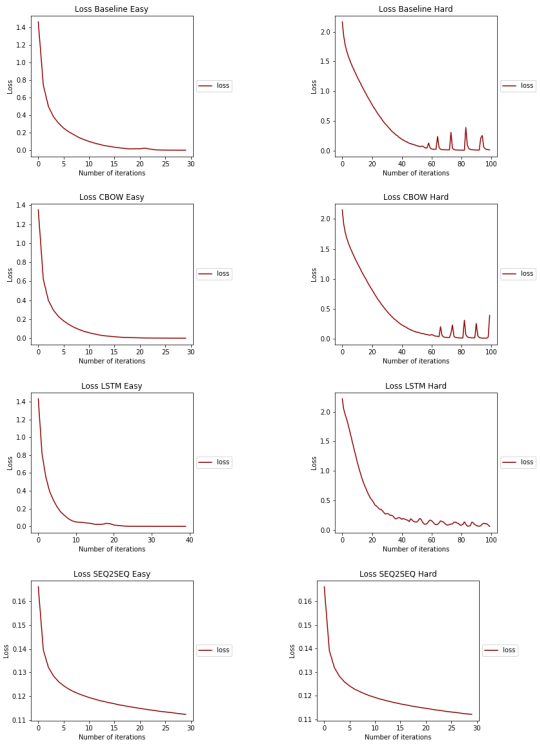

Figure 10: loss plots for four different models