Internship Progress Report

Le-Vu Tran

August 25, 2018

Contents

Chapter 1

Week 11 - August 16 2018

1.1 Introduction

I have spent the majority of my time this week trying my Long - Short Term Memory network with features extracted from Places365's pre-trained ResNet18, learning how to implement LSTM in TensorFlow.

1.2 Places365 Caption

This week I also run Places365 on images to analyze scene categories and attributes. And I concatenated 10 most relevant attributes structuring caption for a concrete image. Finally we tried to used that caption as input of any Convolutional Neural Network to predict images' memorabilities. I rewrote the source code and it could be found here on my GitLab.

1.3 TensorFlow LSTM Classification Network

1.3.1 Architecture

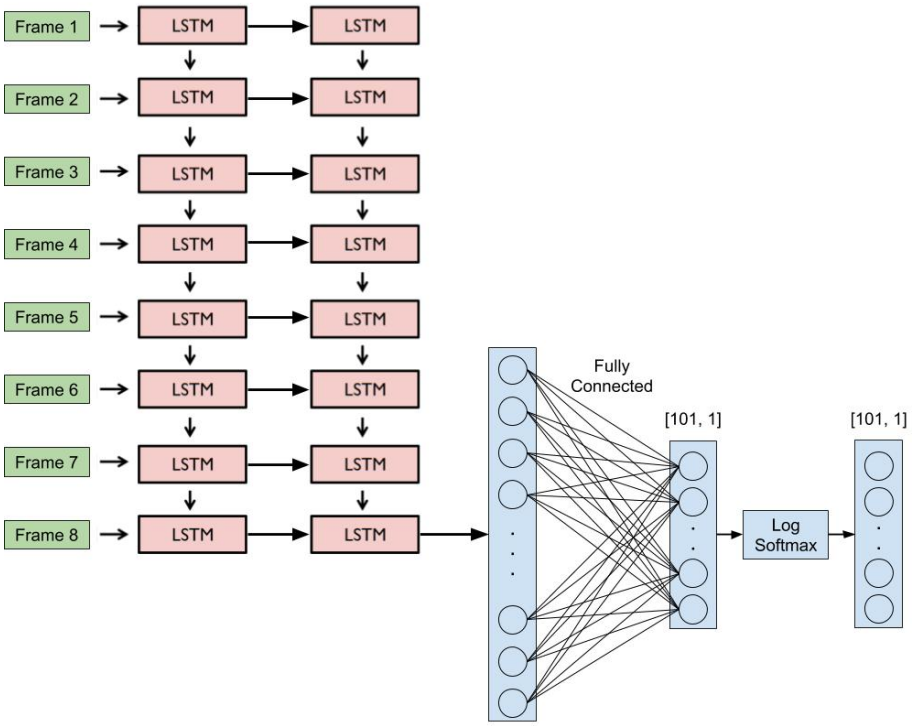

Figure 1.1: Long - Short Term Memory Network Architecture.

This week I tried to implement Long - Short Term Memory network using TensorFlow library. I finally figured out what was the definition of graph and session term in TensorFlow. I also learnt some new TensorFlow functionalities

for example Graph, Session, DropoutWrapper, BasicLSTMCell, MultiRNNCell, AdamOptimizer and argmax.

I re-designed my LSTM network using TensorFlow library and the figure below would demonstrate clearly how my network was. I used a stack of 2 LSTM cells sequences and only took the last LSTM cell's output for my Fully - connected layer. In each version I changed some hyper parameters to figure out whether I could increase the accuracy of not.

1.3.2 Training on Dev-Set (extracted by ResNet50)

Strategy

This times I broke the problem into the classification problem of 101 categories. Because I thought this dataset was pretty small for deeplearning so I splitted the provided Dev-Set for this challenge into two parts, since the Dev-Set had 8000 videos, I picked ***7000*** videos for training and ***1000*** videos for testing.

Version 1

In this version, I trained a batch of size ***128*** at once, with ***512*** LSTM units for each LSTM cell and ***7000*** iterations.

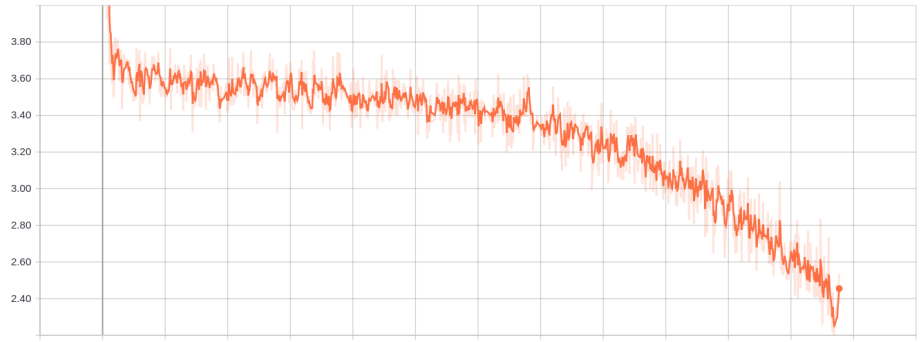

Figure 1.2: Loss over Epoch.

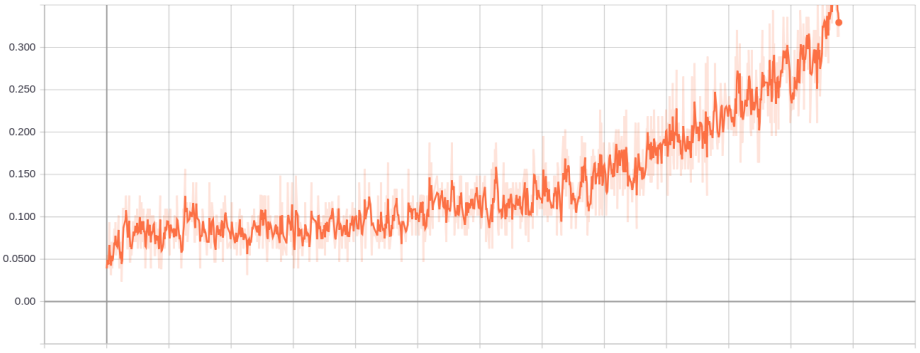

Figure 1.3: Accuracy over Epoch.

After 7000 iterations, the loss value was ***2.5*** and the accuracy value was ***0.3***. But from my perspective view, the loss and accuracy were both turbulent so lately I moved to another version with some hyper-parameter changed. After performing test on the ***train set***, I got the Spearman Rank Correlation of ***0.37***; on ***test set***, the Spearman Rank Correlation was just ***0.05***.

Version 1.1

In this revamp version, I trained a batch of size ***512*** at once, with ***1024*** LSTM units for each LSTM cell and ***7000*** iterations.

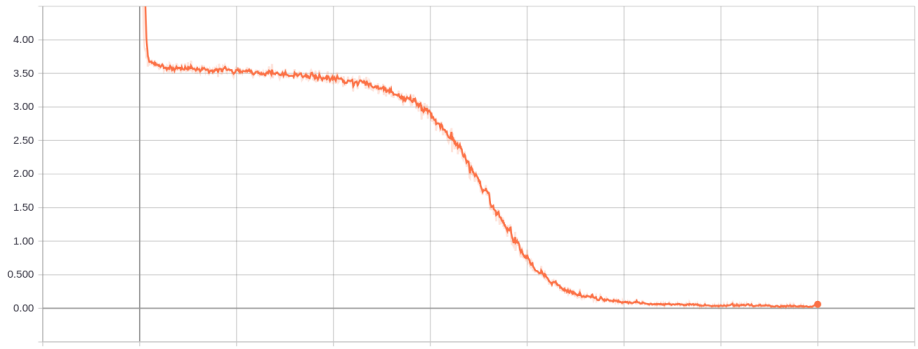

Figure 1.4: Loss over Epoch.

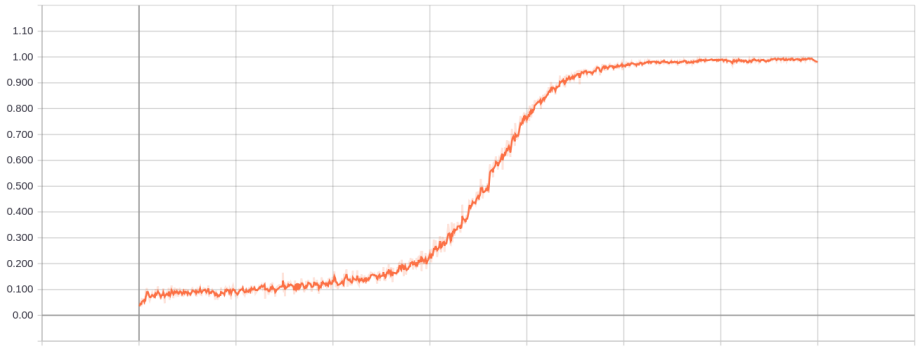

Figure 1.5: Accuracy over Epoch.

After 7000 iterations, the loss value was ***0.06*** and the accuracy value was ***0.98***. But from my perspective view, the loss and accuracy were both turbulent so lately I moved to another version with some hyper-parameter changed. After performing test on the ***train set***, I got the Spearman Rank Correlation of ***0.98***; on ***test set***, the Spearman Rank Correlation was just ***0.03***. This result seemed to be the consequence of overfitting. I thought 7000 iterations was not a good number.

1.3.3 Training on Dev-Set (extracted by Places365)

Strategy

I also broke the problem into the classification problem of 101 cate-gories. I first passed all of my input through Places365's pre-trained ResNet18 network and used those features as input of my LSTM network. The features extracted by ResNet18 had the dimension of ***512***. I aslo picked ***7000*** videos for training and ***1000*** videos for testing too.

Version 1

In this version, I trained a batch of size ***128*** at once, with ***256*** LSTM units for each LSTM cell and ***7000*** iterations.

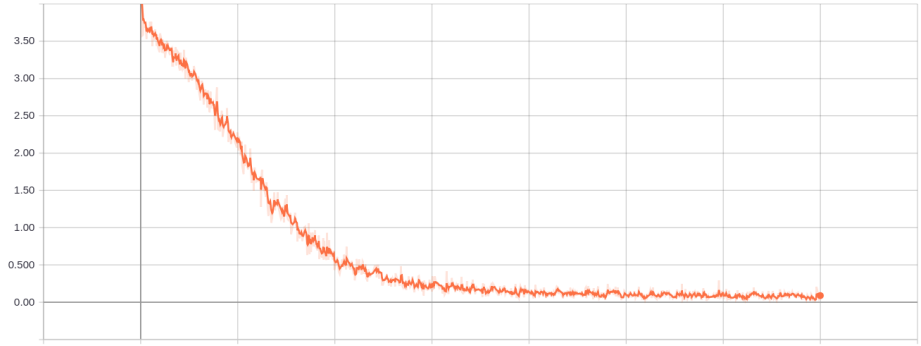

Figure 1.6: Loss over Epoch.

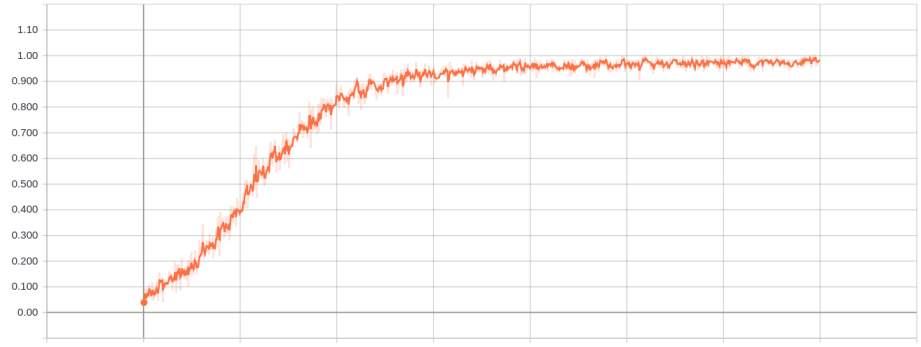

Figure 1.7: Accuracy over Epoch.

After 30000 iterations, the loss value was ***0.09*** and the accuracy value was ***0.98***. But from my perspective view, the loss and accuracy were both turbulent so lately I moved to another version with some hyper-parameter changed. After performing test on the ***train set***, I got the Spearman Rank Correlation of ***0.97***; on ***test set***, the Spearman Rank Correlation was just ***0.07***. This result also seemed to be the consequence of overfitting. I should decrease the number of iterations in the next version.

Chapter 2

Week 11 - August 23 2018

2.1 Introduction

I have spent the majority of my time this week testing the Spearman Rank correlation, Topological idea and grasping the idea of AMNet[66] by Jiri Fajtl.

2.2 Spearman Rank Correlation Experiments

This week I tested the Spearman Rank Correlation computation on groundtruth and some values inherited from the groundtruth itself.

Keep the Groundtruth by itself. I calculated the correlation between the groundtruth and itself to validate my reasoning that it would be 1. These were the results; ***Spearman's Correlation*** value of ***1.0***, ***Pearson's Correlation*** value of ***1.0***, ***Mean Squared Error*** value of ***0.0***.

Divide the Groundtruth by 2. I divided the groundtruth by 2 and calculated the correlation between them. These were the results; ***Spearman's Correlation*** value of ***1.0***, ***Pearson's Correlation*** value of ***1.0***, ***Mean Squared Error*** value of ***0.16***.

Ordinal Ranking. For example I had 6 elements ***[0.45, 0.25, 0.1, 0.2, 0.25, 0.15]***, the ordinal ranking of these 6 elements would be ***[6, 4, 1, 3, 5, 2]***. This ranking technique did not care about equal values, it just ranked the elements by their values and their index (for equal values). I calculated the correlation between the groundtruth and its ordinal ranking. These were the results; ***Spearman's Correlation*** value of ***0.999***, ***Pearson's Correlation*** value of ***0.966***, ***Mean Squared Error*** value of ***21319943.72***. Because the ranks were far from the groundtruth so it was reasonable for the mean squared error value to be a huge number.

Max Ranking. Given 6 elements as the example above, the max ranking of these 6 elements would be ***[6, 5, 1, 3, 5, 2]***. This ranking technique based on the ordinal ranking, but it treated equal values as the same ranks and their ranks would be the maximum one in term of their ordinal ranks. I calculated the correlation between the groundtruth and its max ranking. These were the results; ***Spearman's Correlation*** value of ***1.0***, ***Pearson's Correlation*** value of ***0.968***, ***Mean Squared Error*** value of ***22609763.39***. As this ranks also made the predict results completely differed from the groundtruth so the mean squared error was also a huge number.

Dense Ranking. Given 6 elements as the example above, the dense ranking of these 6 elements would be ***[5, 4, 1, 3, 4, 2]***. This ranking technique treated equal values as the same ranks and kept moving on without making any gaps in the ranking sequence. I calculated the correlation between the groundtruth and its dense ranking. These were the results; ***Spearman's Correlation*** value of ***1.0***, ***Pearson's Correlation*** value of ***0.992***, ***Mean Squared Error*** value of ***12969.23***. Because this ranking technique did not create any gaps so it was much more efficient (in term of memory) and the mean squared error was also smaller than the 2 ranking techniques' above.

After testing these ranking techniques I had a conclusion that the ***Dense Ranking*** would work best because its Pearson's Correlation value was higher and mean squared error value was much smaller than 2 other ranking techniques above. So it would be a reasonable choice for the Topological Sort idea which would be presented next.

2.3 Topological Sort Network

2.3.1 Architecture

This was actually my mentor's idea and I also confirmed its feasibility after experiments on the Spearman Rank Correlation computation above. My network took in 2 videos at once, concatenated them and pass through several Fully-connected layers to get determine which video was more memorabible than the other one. After comparing all two-video pairs I would use Topological Sort algorithm to retrieve the ranking sequence. Spearman Rank Correlation would be calculated on this ranking sequence and the groundtruth.

2.3.2 Training on Dev-Set (extracted by ResNet50)

Strategy

I splitted the provided Dev-Set for this challenge into three parts, since the Dev-Set had 8000 videos, I picked ***6000*** videos for training, ***1000*** videos for validating and the last ***1000*** videos for testing.

Version 1

In this version I used 2 hidden layers which size were ***2048*** and ***128***, I also added 3 dropout layers (probability of ***0.75*** each) between each Fully-connected layers. The figure below demonstrated my version 1 network.

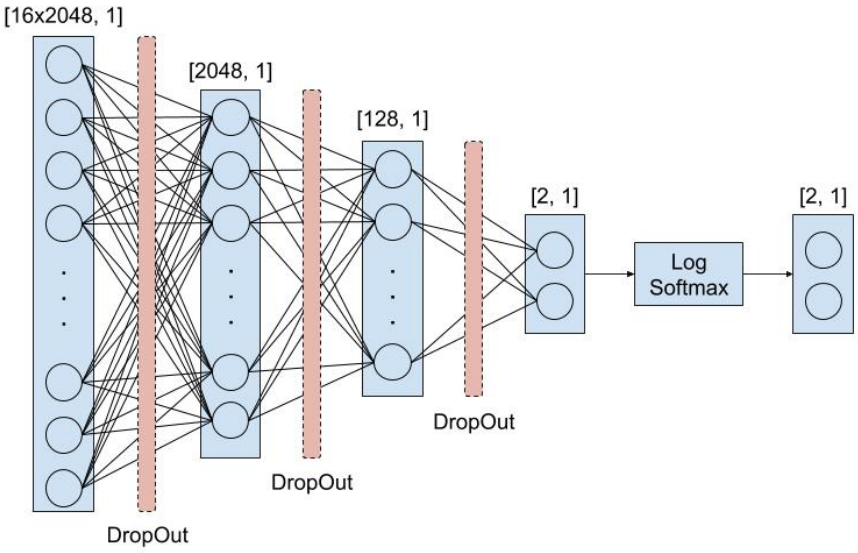

Figure 2.1: Topological Sort Network Network Architecture.

I trained this network with ***1000*** epochs. Each epoch I randomly picked a batch of ***128*** videos and compared each video with others (it meant I had average ***16384*** comparisons each epoch). The loss value after 1000 epochs was ***0.078*** and the highest validate accuracy was ***54.68%***.

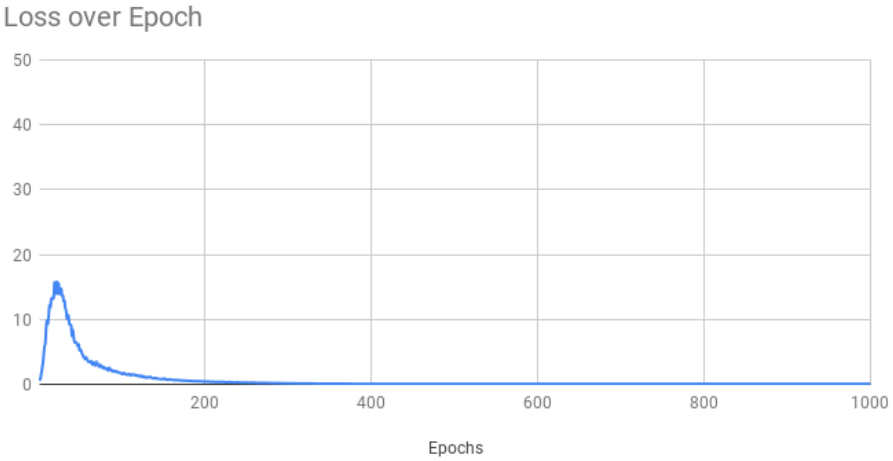

Figure 2.2: Loss over Epoch (on Train set).

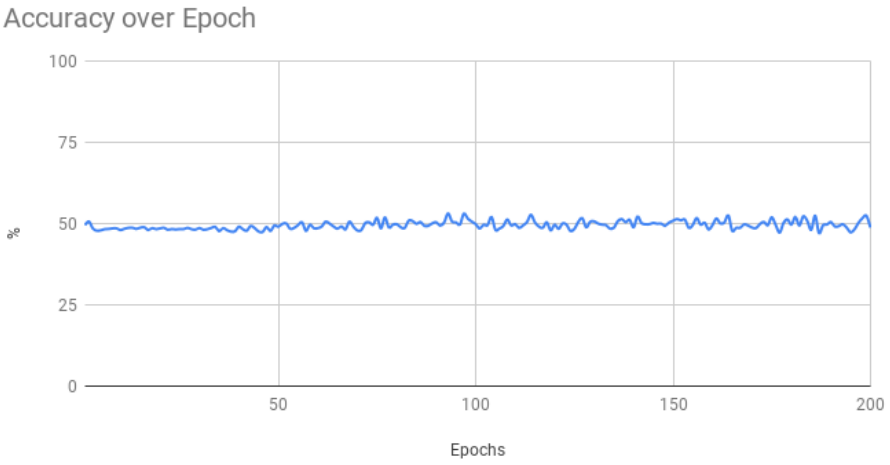

Figure 2.3: Accuracy over Epoch (on Validate set).

Then I used this trained model to calculate all comparisons between videos and used those values for Topological Sort algorithm to get the final ranking sequence. But the Spearman Rank correlation value was only ***0.01***. So I thought this idea was not a reasonable approach.

2.4 AMNet

2.4.1 Introduction

In this paper the authors presented the design and evaluation of an end-to-end trainable, deep neural network with a visual attention mechanism for memorability estimation in still images. The author analyzed the suitability of transfer learning of deep models from image classification to the memorability task. Further on the authors studied the impact of the attention mechanism on the memorability estimation and evaluate their network on the SUN Memorability and the LaMem datasets. Their network outperforms the existing state of the art models on both datasets in terms of the Spearman's rank correlation as well as the mean squared error, closely matching human consistency.

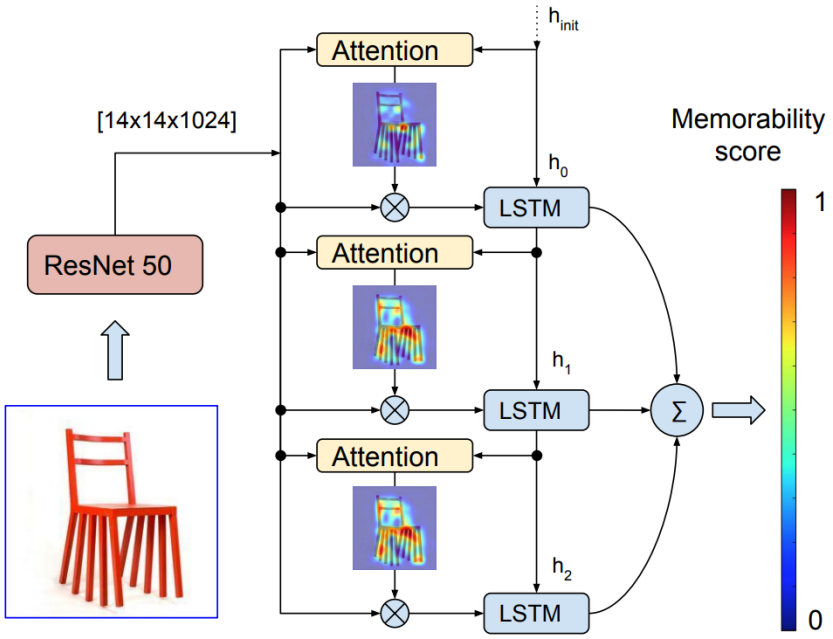

Figure 2.4: AMNet iteratively generates attention maps linked to the image regions correlated with the memorability. After three iterations the memorability scores were added and presented on the output.

The AMNet estimated the image memorability by taking a single image $\boldsymbol{X}$ and generating a memorability score $\boldsymbol{y}$. The process of memorability estimation was summarized in algorithm presented in the figure above. The authors proposed all required mathematical equations (Eq.3, Eq.4, Eq.6, ...) in their paper for implement purpose. The figure below demonstrated the detailed process of memorability estimation.

procedure MEMORABILITY(X) ▷ $y = f(\boldsymbol{X})$
  $\mathbf{x} = get\_cnn\_features(\boldsymbol{X})$ ▷ ResNet50 fwd pass
  $\mathbf{h}_0 = f_{init_c}(\mathbf{x})$ ▷ Eq. 12
  $\mathbf{c}_0 = f_{init_h}(\mathbf{x})$ ▷ Eq. 12
  $lstm\_init(\mathbf{h}_0, \mathbf{c}_0)$
  $y = 0$
  for $t = 0$ to T **do** ▷ at $t = 0 \rightarrow \mathbf{h}_t = \mathbf{h}_0$
    $\mathbf{e} = f_{att}(\mathbf{x}, \mathbf{h}_t)$ ▷ Eq. 8
    $\boldsymbol{\alpha} = softmax(\mathbf{e})$ ▷ Eq. 6
    $\mathbf{z} = []$
    for $i = 0$ to L **do** ▷ for all locations, Eq. 4
      $\mathbf{z} = \mathbf{z} + \alpha_i \mathbf{x}_i$ ▷ $\mathbf{z} \in \mathbb{R}^D$
    $\mathbf{h}_t, \mathbf{c}_t = lstm\_step(\mathbf{z}, \mathbf{h}_t, \mathbf{c}_t)$ ▷ Eq. 3
    $y = y + f_m(\mathbf{h}_t)$ ▷ Eq. 11
  return y ▷ Memorability score $[0, 1]$

Figure 2.5: Summarized AMNet Algorithm.

The authors also proposed their custom loss function in their paper. Their loss function composed of two terms. The first term represented the mean squared error between the groundtruth and predicted image memorability. In order to encourage the attention model to explore all image regions over all time steps, the authors added a second term which performed a joint penalty as a function of activations of all attention maps in the LSTM sequence, which was already introduced by Kelvin Xu[67].

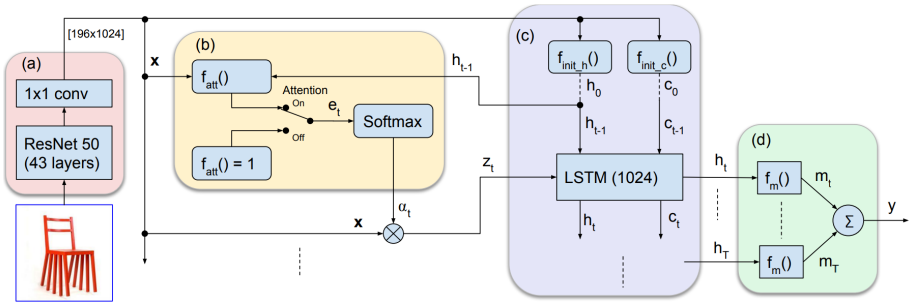

Figure 2.6: Detailed AMNet Algorithm.

The authors claimed that the entire model was fully differentiable and trained end-to-end with the ADAM[68] optimizer with a fixed learning rate 10^3. The input image feature vector was extracted from the 43^{rd} layer of the RestNet50 with dimensions $[14 \times 14 \times 1024]$. The ResNet50 was trained for image classification on the ImageNet dataset and its weights were not updated during the AMNet training.

2.4.2 Training on LaMem-Set (extracted by Inceptionv3)

Strategy

Instead of using the input image feature vector of dimensions $[14 \times 14 \times 1024]$ and having the sequence length of 14×14 as AMNet, I used the input image feature vector of dimensions $[1 \times 2048]$ and my sequence length was 1. I

did not want to complicate my loss function, I just used the mean squared error only. I splitted the LaMem-Set for into two parts, the total number of images was ***55000***, I picked ***45000*** images for training, and ***10000*** images testing.

Version 1

In this version I had my input size of ***2048***, hidden size of ***1024***, ***3*** stacking LSTM layer and as mentioned above, my sequence length of ***1***. My ADAM optimizer's learning rate was ***1e-4***. I trained this version with ***200*** epochs and saved the snapshot which had the highest correlation value on test set. The loss and the correlation both behaved stably.

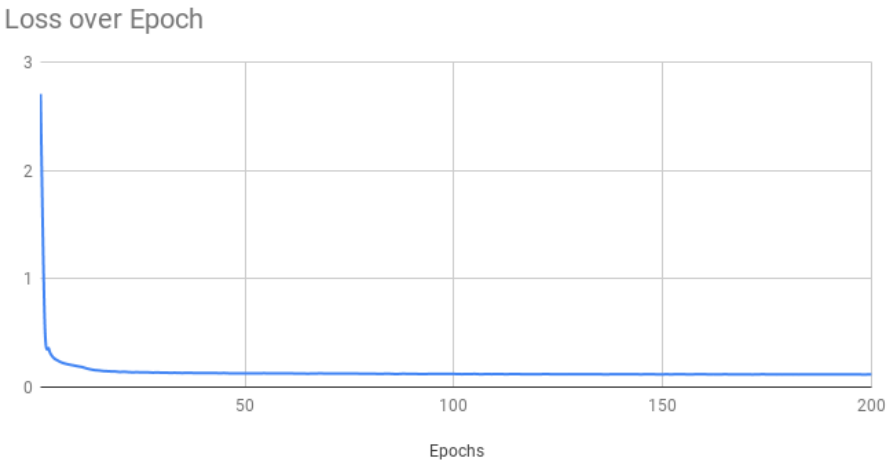

Figure 2.7: Loss over Epoch (on Train set).

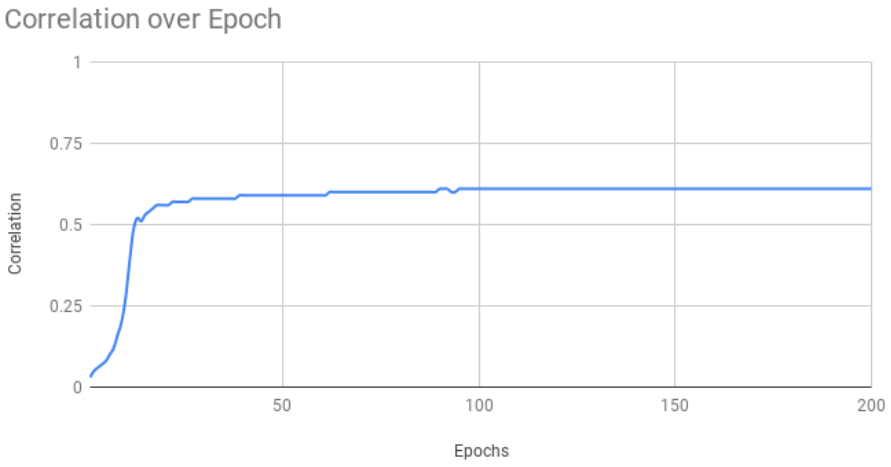

Figure 2.8: Correlation over Epoch (on Test set).

The loss and correlation value stopped significantly changing from the epoch 100^{th} so I thought it only needed 100 epochs to converge. After 200 epochs, my loss was about ***0.117*** and my final correlation value on test set was ***0.61***.

Later, I used this pre-trained model to calculate the correlation value on the Dev-Set but predictably, the result was absolutely terrible, it was not even a positive number.

2.4.3 Training on Dev-Set (extracted by ResNet50)

Strategy

I used the same input strategy as the previous version that I trained on LaMem-Set. I also try to conduct the same loss function as the one proposed by the authors which was a combination of mean squared error and attention penalty. I splitted the provided Dev-Set for this challenge into three parts, since the Dev-Set had 8000 videos, I picked ***6000*** videos for training, ***1000*** videos for validating and the last ***1000*** videos for testing.

2.4.4 Version 1

I kept the input size and hidden size as the same as the previous version that I trained on LaMem-Set. This times I changed the number of stacking LSTM layer to ***1*** and the sequence length to ***8***. There was one new hyper-parameter gamma which specified the impact of the attention penalty. I used its value of ***1e-3***. Finally was the ADAM optimizer with learning rate of ***1e-4***.

After

Bibliography

[1] École Polytechnique Fédérale de Lausanne, *EE-559 - Deep Learning.*

[2] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, A. Torralba, *Places: A 10 million Image Database for Scene Recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017.

[3] Jia, Yangqing and Shelhamer, Evan and Donahue, Jeff and Karayev, Sergey and Long, Jonathan and Girshick, Ross and Guadarrama, Sergio and Darrell, Trevor, *Caffe: Convolutional Architecture for Fast Feature Embedding*, arXiv:1408.5093, 2014.

[4] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, *PyTorch.*

[5] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks.*

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, *Deep Residual Learning for Image Recognition*, arXiv:1512.03385, 2015.

[7] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger, *Densely Connected Convolutional Networks*, arXiv:1608.06993, 2016.

[8] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba, *Learning Deep Features for Discriminative Localization*, arXiv:1512.04150, 2015.

[9] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, *Going Deeper with Convolutions*, arXiv:1409.4842, 2014.

[10] École Polytechnique Fédérale de Lausanne *EE-559 - Deep Learning*, *Mini-project 1: Prediction of finger movements from EEG recordings.*

[11] Karen Simonyan, Andrew Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*, arXiv:1409.1556, 2014.

[12] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, *You Only Look Once: Unified, Real-Time Object Detection.*

[13] Ross Girshick, Jeff Donahue, Trevor Darrell, Jitendra Malik, *Rich feature hierarchies for accurate object detection and semantic segmentation*, arXiv:1311.2524, 2013.

[14] G. C. Benjamin Blankertz and K.-R. R. Muller, *Towards brain computer interfacing*, *Neural Information Processing Systems (NIPS)*, 2002.

[15] Davis E. King, *DLib*.

[16] Stanford University, CS231n - Convolutional Neural Networks for Visual Recognition.

[17] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. *CIFAR-10 dataset*.

[18] Sergey Ioffe, Christian Szegedy *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*.

[19] Conference on Computer Vision and Pattern Recognition 2017.

[20] International Conference on Computer Vision 2017, *2017.thecvf.com*.

[21] Conference on Computer Vision and Pattern Recognition 2018.

[22] Eric Tzeng, Judy Hoffman, Kate Saenko, Trevor Darrell, *Adversarial Discriminative Domain Adaptation*, arXiv:1702.05464, 2017.

[23] Artem Rozantsev, Mathieu Salzmann, Pascal Fua, *Beyond Sharing Weights for Deep Domain Adaptation*, arXiv:1603.06432, 2016.

[24] Mehdi Mirza, Simon Osindero, *Conditional Generative Adversarial Nets*, arXiv:1411.1784 2014.

[25] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, Dimitris Metaxas, *Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks*, arXiv:1612.03242, 2016.

[26] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, Dimitris Metaxas, *StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks*, arXiv:1710.10916, 2017.

[27] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, *Generative Adversarial Networks* arXiv:1406.2661, 2014.

[28] Han Zhang, Ian Goodfellow, Dimitris Metaxas and Augustus Odena, *Self-Attention Generative Adversarial Networks*, arXiv:1805.08318, 2018.

[29] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, Yuichi Yoshida, *Spectral Normalization for Generative Adversarial Networks*, arXiv:1802.05957, 2018.

[30] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros, *Paired Image-to-image translation using Conditional GAN*, arXiv:1611.07004, 2016.

[31] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro, *High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs*, CVPR 2018.

[32] Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros, *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*, ICCV 2017.

[33] Zhu, Jun-Yan and Zhang, Richard and Pathak, Deepak and Darrell, Trevor and Efros, Alexei A and Wang, Oliver and Shechtman, Eli, *Toward Multimodal Image-to-Image Translation*, arXiv:1711.11586, Nov 2017.

[34] Xiaolong Wang, Abhinav Shrivastava, Abhinav Gupta, *A-Fast-RCNN: Hard Positive Generation via Adversary for Object Detection*, CVPR, 2017.

[35] Ross Girshick, *Fast R-CNN*, arXiv:1504.08083, 2015.

[36] Jun-Yan Zhu, *pytorch-CycleGAN-and-pix2pix*.

[37] Cameron Smith, *neural-style-tf*.

[38] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, *Image Style Transfer Using Convolutional Neural Networks*.

[39] Manuel Ruder, Alexey Dosovitskiy, Thomas Broxn, *Artistic style transfer for videos*, arXiv:1604.08610, 2016.

[40] Leon A. Gatys, Matthias Bethge, Aaron Hertzmann, Eli Shechtman, *Preserving Color in Neural Artistic Style Transfer*, arXiv:1606.05897, 2016.

[41] The Starry Night by Vincent van Gogh, 1889.

[42] Multimedia Evaluation 2018, *mediaeval2018*.

[43] Medico: The 2018 Multimedia for Medicine Task.

[44] Joint Photographic Experts Group.

[45] The 2018 Emotional Impact of Movies Task, *emotionalimpact*.

[46] The 2018 Predicting Media Memorability Task, *memorability*.

[47] James L McGaugh, *Memory-a Century of Consolidation*, Science 287, 5451 (2000), 248-251.

[48] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, *Learning Spatiotemporal Features with 3D Convolutional Networks*, ICCV 2015.

[49] Khurram Soomro, Amir Roshan Zamir and Mubarak Shah, *UCF101: A Dataset of 101 Human Action Classes From Videos in The Wild*, CRCV-TR-12-01, November 2012.

[50] ALMEIDA, Jurandy; LEITE, Neucimar J.; TORRES, Ricardo da S, *Image Processing (ICIP), 2011 18th IEEE International Conference on*. IEEE, 2011. p. 3673-3676.

[51] Dalal, N. and B. Triggs, *Histograms of Oriented Gradients for Human Detection*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1 (June 2005), pp. 886–893.

[52] DC. He and L. Wang (1990), *Texture Unit, Texture Spectrum, And Texture Analysis*, Geoscience and Remote Sensing, IEEE Transactions on, vol. 28, pp. 509 - 512.

[53] SZEGEDY, Christian et al, *Rethinking the inception architecture for computer vision*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016. p. 2818-2826.

[54] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski, *Orb: an efficient alternative to sift or surf*, In Computer Vision (ICCV), 2011, IEEE International Conference on, 2011.

[55] OpenCV *Keypoints and Descriptors tutorial*.

[56] Eugenio Culurciello, TowardsDataScience, *Neural Network Architectures*.

[57] Sepp Hochreiter; Jürgen Schmidhuber, *Long Short-Term Memory*, Neural Computation.

[58] Google's AI organization, *TensorFlow*.

[59] Ronan Collobert, Koray Kavukcuoglu, Clement Farabet, *Torch*.

[60] Skymind, *From word to embeddings*.

[61] Aditya Khosla, Akhil S. Raju, Antonio Torralba and Aude Oliva, *Understanding and Predicting Image Memorability at a Large Scale*, International Conference on Computer Vision, 2015.

[62] Matt Harvey, Coastline Automation, *Five video classification methods implemented in Keras and TensorFlow*.

[63] Phillip Isola, Jianxiong Xiao, Antonio Torralba and Aude Oliva, *What makes an image memorable?*, IEEE Conference on Computer Vision and Pattern Recognition, 2011.

[64] T. Konkle, T. F. Brady, G. A. Alvarez, and A. Oliva, *Conceptual distinctiveness supports detailed visual long-term memory for realworld objects*, JEP:G, 2010.

[65] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, *LabelMe: a database and web-based tool for image annotation*, IJCV, 2008.

[66] Jiri Fajtl, Vasileios Argyriou, Dorothy Monekosso, Paolo Remagnino, *AMNet: Memorability Estimation with Attention*, arXiv:1804.03115, 2018.

[67] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, *Show, Attend and Tell: Neural Image Caption Generation with Visual Attention*, arXiv:1502.03044, 2015.

[68] Diederik P. Kingma, Jimmy Ba, *Adam: A Method for Stochastic Optimization*, arXiv:1412.6980, 2014.