2AMM10: Assignment 1

Andrei Rykov Nikita Jain Ryan Meghoe

May 23, 2022

1 Problem formulation

Classification is one of the most common problems of machine learning. Over time there are multiple solutions provided to solve it. However, there exists a specific task of classification known as one-shot learning. In one-shot learning setting the model should be able to learn the class from only few examples, and the resulting model should distinct the objects from variety of different classes with only a few examples of each class given for training.

Solution for that problem is the metric learning. Instead of trying to classify each object separately and then compare classes to find out if two object are in the similar class we'll learn the representation f of the image X in some m-dimensional metric space $\mathbb{R}^m$ and then learn the threshold τ value for some chosen distance metric $d(X_1, X_2) = ||f(X_1) - f(X_2)||$ such that $d(X_1, X_2) > \tau$ for two images X_1, X_2 from different classes, and $d(X_1, X_2) < \tau$ for X_1, X_2 from the same class. Such representation learns some important features of images from the specific domain so it can distinct two objects from classes it has never observed before (but classes belong to the same domain as the train data). In addition, we should classify only the object wherever it is located in the picture, so the model should be invariant to the simple affine transformations such as translation (the location of the object in the picture), rotation and scale. The report focuses on the implementation of the model, which learns a good representation of characters from the Omniglot dataset with images of size 28x28. Furthermore, the model learns the τ such that it aids in the prediction if two input characters are not observed before. The goal is to predict if images from the support set of size 5 belong to the same class as the query image. As the number of characters that belong to the query image's class can vary, the task can be simplified to the pairwise classification of each image from the support set and the query image.

2 Model formulation

The Siamese network in [1] is applied to tackle this problem. The network is known to be quite successful in the one-shot learning setting. Due to too many different classes in the dataset, the approach should be applicable to our problem. The basic idea is to pass two incoming images into a convolutional neural network in parallel. That neural network would extract most relevant features and create corresponding embeddings of input images. The embeddings are then substracted from each other. This difference represents the "distance" or similarity of two given images. The difference is then used in fully connected layers to make a prediction.

Thus, the whole model can be represented as:

$$f(x_q, x_i) = f_2(|f_1(x_q) - f_1(x_i)|)$$

,

where f_1 - is the convolutional part of the neural net, which outputs the embedding, and f_2 - is the fully connected part. The output is defined as the probability that two images belong to the same class.

The convolutional Net that we use for the embedding extraction is also known to be invariant to the affine transformations, so the resulting representation of the image will store information about the object in it indifferently to the location, scale, and position of the object.

For the optimisation the binary cross-entropy loss function was chosen [1]:

$$L(x_q, x_i) = \mathbb{1}\{x_i, x_q\} f(x_q, x_i) + (1 - \mathbb{1}\{x_i, x_q\})(1 - f(x_q, x_i))$$

To make the model fit our problem, the dataset was transformed for the task of binary classification: each support set of images $\{x_i\}$ for i $\in$ [0..4] and query image x_q is transformed into array of tuples $\{((x_i, x_q), y_i)\}$ for i $\in$ [0..4], where $y_i = \mathbb{1}\{x_i$ is the same class as $x_q\}$ - characteristic function (later will refer to this function as $\mathbb{1}\{x_i, x_q\}$). In the original problem statement, we should have been able to classify whether each character in support set is from the same class as the character from the query image. Thus, we can simplify this problem into the pairwise comparison of each image from support set and the query image.

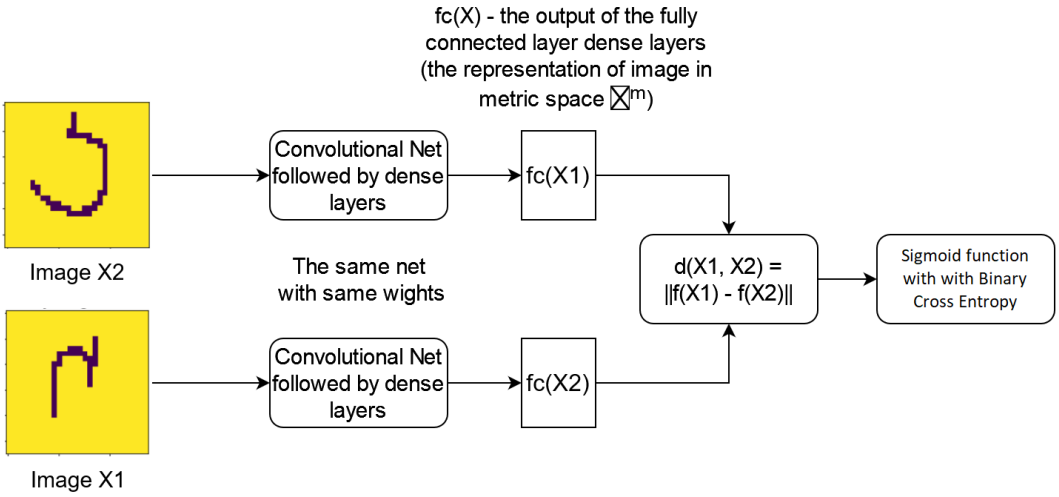

Figure 1: The general depiction of the Siamese Neural Network architecture

3 Implementation

In order to train the Siamese model each image is stored as a pair. The formation of the pairs is achieved by combining an image of the support vector, and one from the query set from a different class. Hereafter, for the training and validation, the random length of the classes is multiplied by the train-size. The value assigned to the train-size is 80%.

As for the test, train, and validation data loaders, a batch size of 10 is assigned. As for the construction of the Siamese model described in Section 2, we used three convolution layers, each with a kernel size of 3x3. In the first convolution layer, the input number of channels is 1 (as the

original images are black and white one-channel images with size of 28x28 pixels), with an output numer of channels equal to 64, while in the second layer the output channels increase up to 128 and the input size equals the output size of layer one. The last layer of the model returns the output with 256 channels. Additionally, after each layer, a rectified activation layer is applied with average pooling with a kernel size of 2 (except the third convolutional layer) followed by batch normalization. Eventually, the output is flattened into vector of size 2304 (3x3 output of 256 various channels) and passed to linear layer with sigmoid activation function that produces embedding of size 128. After that, the vector of absoulute difference of two embeddings - the query image and the image from support set - is computed and passed to the last linear layer with sigmoid function. As the result, the model outputs the probability of two input images be in the same class.

Layer	Type
1	Conv2d(in_channels = 1, out_channels = 64, kernel_size=(3, 3))
-	ReLU()
2	MaxPool2d(kernel_size=2)
3	BatchNorm2D(channels = 64)
4	Conv2d(in_channels = 64, out_channels = 128, kernel_size=(3, 3))
-	ReLU()
5	MaxPool2d(kernel_size=2)
6	BatchNorm2D(channels = 128)
7	Conv2d(in_channels = 128, out_channels = 256, kernel_size=(3, 3))
-	ReLU()
8	BatchNorm2D(channels = 256)
9	Linear(in_features=2304, out_features=128, bias=True)
-	Sigmoid()
10	Linear(in_features=128, out_features=1, bias=True)
-	Sigmoid()

Table 1: Description of implemented Siamese Network with Convolutional Net.

Regarding training, the binary-cross entropy loss is used,which compares the predicted probabilities to actual class output. The actual class outputs can be either 0 or 1.Following the comparison, a score is calculated that penalizes probability based on their deviation from the expected value.

4 Experiments

For experimenting, we used the train and test parts of the Omniglot dataset. The dataset was divided into a training dataset (80% of the training dataset) and a validation dataset (20% of the training dataset). To both parts augmentation was applied: to both images in the input tuple, the random affine transformation was applied using PyTorch Module RandomAffine. Each picture could have been rotated, translated, and (or) scaled: the range of degrees for the rotation was $(-15°, 15°)$, and the shift distance was up to the 30% of the image width and (or) height, each image could have been scaled up by 15% of its' size or scaled-down by 10%. All described transformations were not applied to the test set.

In addition, we compared the performance of our model with the modified version for Contrastive Loss function [2], which can be written as:

Model	Train	Validation	Test
Siamese-Net + Original data	**1.**	**0.9401**	0.9050
Siamese-Net + augmentation	0.8650	0.8802	**0.9332**
Siamese-Net with Contr. Loss + augmentation	0.6050	0.6120	0.6284

Table 2: Results of the experiments (accuracy on various sets).

$$L(x_q, x_i) = \mathbb{1}\{x_i, x_q\}\frac{1}{2}||f_1(x_q) - f_1(x_i)|| + (1 - \mathbb{1}\{x_i, x_q\})\max(0, \tau - ||f_1(x_q) - f_1(x_i)||)$$

, where f_1 - convolutional part of the siamese network, which has embeddings as an output; τ - learnable margin parameter. To apply the contrastive loss, the model was adapted and new learning parameter was added to the model - the threshold τ, while last linear layer with sigmoid function was removed from the model. As the result, the edited model outputs the distance between two embeddings and compares it with parameter τ to find if two input images are in the same class.

Since the prepared dataset has a balanced distribution of labels, only the accuracy metric was used for the evaluation.

5 Results and analysis

We obtained the results provided in the 2. As we can see, Siamese Network trained on augmented data with binary cross-entropy as loss showed the best results out of all tested models. Augmentation seems to provide more generality to the model, so it can be easier to distinguish different characters from each other. However, the best test accuracy of our Siamese Model is 0.93 and it seems to be pretty good (as the maximum is 1), we can see that [3] results on the original data are worse by Simple ConvNet. However, almost all models shows higher accuracy rate (lower classification error rate which is equal to 1 - accuracy) using the augmented data: most of the recent models show accuracy > 0.95, that indicates that we did not fully use the potential of data augmentation in our model. Besides that, we can see that training and validation accuracy on the original data is the highest though the result on test set is not the best. We can conclude that mode learns features that are more specific to the train data, but cannot be expanded to the other characters it has never seen before. In addition, we need to mention that we do not know from the given data if the original combinations of support sets and query images are from the same alphabet or from distinct alphabet, so provided comparison is not precise.

Another disappointment is the poor performance of the contrastive loss used for the training of our model: the accuracy of that model was a bit higher than a random guess - 0.6284. We should conclude that either loss is not suitable for the given model and task or either there is a mistake in the implementation, and application of the given loss.

6 Conclusion

The Siamese model is applied as the classification model to classify the characters in the omniglot dataset. The application of the model is done on a dataset of 10000 sets of 6 images each. Each set consists of 5 support images and 1 query image. In each set, the first five columns are support images, and the last one is a query image. To make the classification possible first a

combination of pairs is done. After combining pairs from the query and support set. After the datapreparation phase as described above, the model is constructed. In total the Siamese model consist of 4 blocks: 3 of them are convolutional layers combined with pooling layers (first two layers) and batch normalization, the last one consist of two linear layers with sigmoid activation function - 10 layers in total. To further test the validity of our model, also some data augmentation is carried out. Aside from using the binary cross entropy encoder, the contrasistive loss is also applied. Thus, experimenting is done with the siamese model on the formed dataset, an augmented dataset,and different loss method than the binary cross entropy loss.The loss was the contrastive loss. The model yielded the highest accuracy of 0.93. The test accuracy for augmentation is the highest as the model learns more general features and becomes more "immune" to the affine transformations. However, other scientific results with augmentation applied show better results, so we can conclude that we can utilize augmentation in more efficient way and the model can be upgraded as well.

References

[1] Koch, G., Zemel, R. Salakhutdinov, R. (2015). Siamese Neural Networks for One-shot Image Recognition.

[2] R. Hadsell, S. Chopra and Y. LeCun, "Dimensionality Reduction by Learning an Invariant Mapping," 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), 2006, pp. 1735-1742, doi: 10.1109/CVPR.2006.100.

[3] B. Lake, R. Salakhutdinov, J. Tenenbaum, "The Omniglot challenge: a 3-year progress report" Current Opinion in Behavioral Sciences, 2019, 29, pp. 97-10