Exploring MNIST-MIX

Umair Khan, Grant Baker, Mi Yon Kim, Damon Aliomrany, Bach Khuat

Note: All of our code for this project can be found via GitHub.

Introduction

MNIST-MIX [1] is a recently-released set of handwritten digits composed of examples from Arabic, Bangla, Devanagari, English, Farsi, Kannada, Swedish, Telugu, Tibetan, and Urdu. The data has been processed to match the classic MNIST format (28x28 grayscale), and the ~600K examples have been assigned an 80/20 train-test split between 100 different classes (10 languages x 10 digits). Both the train and test sets preserve the same class ratios. The table below summarizes how the data is spread over the languages.

Language	Train	Test	Total
Arabic	62400	10600	73000
Bangla	39990	9150	49140
Devanagari	2400	600	3000
English	240000	40000	280000
Farsi	60000	20000	80000

Language	Train	Test	Total
Kannada	60000	20420	80420
Swedish	6600	1000	7600
Telugu	2400	600	3000
Tibetan	14214	3554	17768
Urdu	6606	1414	8020

The goal of the project was to investigate different neural network models along with the dataset itself by modifying one or both and comparing it to baseline results. We performed four experiments in pursuit of this goal: (a) we compared the raw performance of feedforward and convolutional models; (b) we tested how models learned using language-agnostic (digit-only) labeling; (c) we explored if and how learning could be

transferred by training on one language and testing on others, and (d) we investigated the training process itself by training on languages sequentially instead of all together.

Related Work

The authors of the original MNIST-MIX dataset provided baseline results using a LeNet model, which achieves an overall accuracy of 90.22% [1]. Our convolutional model is able to outperform this metric. (Note that MNIST-MIX was released in April 2020.)

On a broader level, applications of machine learning in a multilingual context are extremely varied. Banea et al. [2] have previously examined multilingual data for subjectivity analysis, and are able to create more effective models as a result. Similarly, multilingual data has played a role in developing machine learning for sentiment analysis [3], [4], as well as in Google's translation system [5].

Standardized data sets across a large number of languages seem to be rarer. Many text corpora provide parallel data between two languages. Furthermore, government documents (such as those from the United Nations or European Union) are often translated into many languages. However, there do not appear to be many small-scale multilingual datasets like MNIST-MIX that cover a wide range of languages while keeping the task relatively simple and the amount of data relatively small.

Methods

The dataset is formatted as an NPZ archive, which is a gzip-compressed NumPy array. Data is split into training and testing images and labels. We preprocessed the data by

first combining both training and testing data into one large set, and then storing the indices of the training and testing examples, split by language. Data loading and shuffling are done based on the indices, rather than the dataset itself.

To create neural networks, we used PyTorch v1.5. We constructed two models, the feedforward (FF) model, and the convolutional model. The convolutional model is based on ResNet architecture [7]. The FF model uses the sigmoid activation function and accepts as parameters the number of hidden nodes in each layer, formatted as a list. On the other hand, the ResNet model uses the ReLU activation function and accepts a list of block sizes corresponding to the number of residual blocks in each section of the network. To simplify nomenclature, we use [m]x[n] to refer to an FF network with m layers of n nodes, and ResNet-[x] to refer to a ResNet with x total convolutional layers.

Given a batch size (16 for FF, 32 for ResNet), we optimize using gradient descent with momentum. Learning rate and momentum are passed as parameters. For FF networks, the learning rate was set to 0.1, and for ResNets, it was set to 0.01. Momentum was set to 0.9 for all networks. After each epoch of training, we run an evaluation on the test data set to see how well the model is performing. After training is complete, we run a final test evaluation and save the confusion matrix and accuracy. FF networks were trained for 50 epochs, and ResNet networks were trained for 150 epochs.

We also implemented the ability to save the trained models, so we could share and reuse models on different data sets (e.g. language-specific, language-agnostic). This helped us save time in training and running our experiments.

Results

We first trained three FF networks on the full dataset (100 classes) and recorded their accuracies: 2x100 had an accuracy of 61.63%, 2x250 had an accuracy of 63.74%, and 2x500 had an accuracy of 62.14%. To compare, we trained 5 ResNets: ResNet-8 had an accuracy of 75.06%, ResNet-14 had an accuracy of 91.30%, ResNet-20 had an accuracy of 93.15%, ResNet-26 had an accuracy of 93.86%, and ResNet-32 had an accuracy of 93.54%.

For the rest of the experiments, we used the 2x250 and ResNet-26 networks since those performed the best. We then trained the models using language-agnostic labels: each training example was simply labeled 0-9 regardless of language. To better compare with our fourth experiment, we trained on each language one epoch at a time, e.g. one epoch of Arabic, then one epoch of Bangla, and so on, repeating the whole process 200 times. The 2x250 network achieved a final accuracy of 88.99%, whereas the ResNet-26 got 71.17%.

Next, we trained 2x250 and ResNet-26 networks on each language and tested them on all other languages. The results are summarized in the below tables, with all numbers representing accuracies expressed as percentages. Table 1 contains data for the 2x250 network, and Table 2 contains the data for ResNet-26. The column represents the training language, and the row represents the testing language.

[Table 1]

Te / Tr	**Arabic**	**Ban.**	**Dev.**	**Eng.**	**Farsi**	**Kan.**	**Swe.**	**Tel.**	**Tib.**	**Urdu**
Arabic	97.17	10.99	12.4	17.36	15.46	2.613	9.632	7.641	5.462	12.22
Ban.	10.4	93.5	15.47	26.17	13.4	11.67	18.85	12.89	17.82	12.4
Dev.	11	25.5	88.66	15	11.66	15.83	17	29.83	21.33	12.83
Eng.	17.4	24.12	16.91	98.95	6.67	17.71	60.63	10.39	17.41	10.74
Farsi	56.6	14.49	3.71	13.11	97.92	6.145	17.24	7.17	8.799	38.66

Kan.	2.638	18	13.82	16.15	7.732	79.7	12.66	10.48	7.579	12.93
Swe.	6.7	14.89	15.7	86.6	3.2	12.1	81	12.3	14.49	9.2
Tel.	3.666	20.83	35.16	11.5	6.166	8.5	17.66	94.16	17.33	5.833
Tib.	5.12	18.42	23.88	22.62	19.1	12.43	23.46	7.034	82.55	0.984
Urdu	49.22	2.121	1.697	16.47	53.39	20.86	17.82	8.769	2.192	79.98

[Table 2]

Te / Tr	**Arabic**	**Ban.**	**Dev.**	**Eng.**	**Farsi**	**Kan.**	**Swe.**	**Tel.**	**Tib.**	**Urdu**
Arabic	99.27	8.264	4.811	24.04	56.73	2.896	18.29	6.537	9.122	24.08
Ban.	14.69	99.33	24.05	25.21	14.51	26.17	26.15	14.26	23.75	10.53
Dev.	12.5	29.16	98.16	30.5	11.16	33.83	30.16	34	35.66	9.5
Eng.	19.58	30.29	16.88	99.76	21.73	34.77	92.23	10.32	13.91	13.11
Farsi	75.94	12.9	12.12	16.7	99.63	7.465	17.03	9.585	17.23	53.52
Kan.	4.085	23.2	16.77	16.72	8.092	90.94	14.03	13.02	16.05	17.48
Swe.	12.1	26.4	13.1	98	12.9	23.59	99.6	10.4	20	12.4
Tel.	5.333	29.33	40.66	25.33	8.666	58.83	25.33	99	17	12.83
Tib.	5.177	24.28	25.63	31.54	23.15	20.79	24.5	11.84	99.24	3.742
Urdu	72.77	4.314	0.424	20.79	67.82	15.62	16.33	6.718	0.565	98.37

Finally, we trained models on each language sequentially. This meant training a model on only Arabic, then only Bangla, then only Devanagari, and so on through all languages. After a model was trained on a language, it didn't see that language again until testing. We used language-agnostic labels for this experiment. The 2x250 network got an accuracy of 29.63% and the ResNet-26 got an accuracy of 35.93%.

Discussion

Based on the first experiment, we can definitively conclude that ResNet architecture is far more powerful than the basic FF network. In addition to outperforming the FF model

by over 30% (ResNet-26 vs. 2x250), the ResNet-26 model had a higher accuracy on each language compared to the 2x250 network, as seen in Tables 1 and 2 (over 98% for everything except Kannada). This demonstrates the power of the convolutional model, which is better able to capture spatial features that are especially useful for tasks like handwritten digit recognition.

Surprisingly, when we started using language-agnostic labels, the 2x250 model outperformed ResNet-26. In fact, the feedforward model did better with the language-agnostic labels than it did with the full label set. It is not entirely clear why this occurred, but it may have something to do with the fact that the 2x250 model is much smaller than the ResNet-26. Larger models are more prone to overfitting -- the ResNet-26 might have been learning "too quickly" in the language epochs to generalize across all digits, whereas the smaller structure of the 2x250 network helped it generalize.

This can be supported by the per-language training data. When the entirety of the training process was dedicated to learning a specific language, the ResNet-26 outperformed the 2x250 model, as would be expected. We can also use the per-language data to gain some insight into the languages themselves. On the next page is a table containing all (typed) digits, roughly color-coded by the similarity of form.

Broadly speaking, the Arabic/Farsi/Urdu, Devanagari/Kannada/Telugu, and English/Swedish languages have similar symbols. We can see this reflected in the data -- both the 2x250 model and ResNet-26 do reasonably well on all the languages in the common set when trained on any one of them. Notably, the "learning transfer" seems to be

Arabic	٠	١	٢	٣	٤	٥	٦	٧	٨	٩
Bangla	০	১	২	৩	৪	৫	৬	৭	৮	৯
Devanagari	०	१	२	३	४	५	६	७	८	९
English	0	1	2	3	4	5	6	7	8	9
Farsi	۰	۱	۲	۳	۴	۵	۶	۷	۸	۹
Kannada	೦	೧	೨	೩	೪	೫	೬	೭	೮	೯
Swedish	0	1	2	3	4	5	6	7	8	9
Telegu	౦	౧	౨	౩	౪	౫	౬	౭	౮	౯
Tibetan	๐	๑	๒	๓	๔	๕	๖	๗	๘	๙
Urdu	۰	۱	۲	۳	۴	۵	۶	۷	۸	۹

stronger for the ResNet model than for the 2x250 model, with lower accuracy drops between languages in a set. This may speak to the convolutional model's reliance on the spatial form. The feedforward model, while still decent, loses much more performance between common languages -- over 30% between Swedish and English. Interestingly, although Arabic, Farsi, and Urdu are quite similar in symbols, the data shows that the learning transfers are much better from Arabic and Farsi than they are from Urdu. This may be due to the small amount of Urdu data present compared to the other languages.

We see that performance drops sharply when we train on each language sequentially compared to mixing languages on a per-epoch basis. Both models drop to extremely low accuracies, and whereas the 2x250 outperformed the ResNet-26 with the per-epoch mixing, it once again drops below the convolutional model (29.63% versus 35.93%). This demonstrates the importance of using a mixed ordering of examples during training. If examples from a certain category are shown once and never again, that information gets "overwritten" by new examples.

We ran into several noteworthy phenomena during the project. Sometimes, we got an unstable result when using mixed data where the accuracy hovered around 3.7% and didn't improve. This appears to be indicative of a particularly strong local minimum, combined with a high learning rate. It shows how the complexity of the data affects the training of the models. We also saw that there are some relationships between the batch size, accuracy, epochs, and the running time. When we set the batch size smaller, the accuracy got higher in fewer epochs, but it also took longer per epoch. Finding the right balance took some time, which is how we arrived at the parameters previously described.

References

[1] W. Jiang, "MNIST-MIX: A Multi-language Handwritten Digit Recognition Dataset," *ArXiv200403848 Cs*, Apr. 2020, Accessed: Jun. 12, 2020. [Online]. Available: http://arxiv.org/abs/2004.03848.

[2] C. Banea, R. Mihalcea, and J. Wiebe, "Multilingual subjectivity: are more languages better?," in *Proceedings of the 23rd International Conference on Computational Linguistics*, Beijing, China, Aug. 2010, pp. 28–36, Accessed: Jun. 12, 2020. [Online].

[3] E. Boiy and M.-F. Moens, "A machine learning approach to sentiment analysis in multilingual Web texts," *Inf. Retr.*, vol. 12, no. 5, pp. 526–558, Oct. 2009, doi: 10.1007/s10791-008-9070-z.

[4] A. Balahur and M. Turchi, "Improving Sentiment Analysis in Twitter Using Multilingual Machine Translated Data," in *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, Hissar, Bulgaria, Sep. 2013, pp. 49–55, Accessed: Jun. 12, 2020. [Online]. Available: https://www.aclweb.org/anthology/R13-1007.

[5] M. Johnson *et al.*, "Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation," *Trans. Assoc. Comput. Linguist.*, vol. 5, pp. 339–351, Dec. 2017,

doi: 10.1162/tacl_a_00065.

[6] D. Elliott, S. Frank, K. Sima'an, and L. Specia, "Multi30K: Multilingual English-German Image Descriptions," *ArXiv160500459 Cs*, May 2016, Accessed: Jun. 12, 2020. [Online]. Available: http://arxiv.org/abs/1605.00459.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.