Optimal network architecture for the identification of manipulated versus unmanipulated images using eye gaze data

James Mills-Thom

Research School of Computer Science
The Australian National University
Canberra, ACT 2600 Australia
E-mail: u6380733@anu.edu.au

Abstract.
In the digital age, it is becoming increasingly easy to manipulate digital images – a common media form seen globally. These manipulations can range from obvious and humorous, to covert and malicious. A fake image posted to the internet can mislead thousands of people, wrongfully informing their decisions and opinions. As such, it is important to be able to discriminate between manipulated and unmanipulated images. The present paper extends upon previous work by the same author [6] and aims to improve classification results on an image dataset [3] by applying deep learning (DL) techniques. It was found that an ensemble method utilizing convolutional neural networks (CNNs) and long short-term memory (LSTM) cells was able to improve accuracy. It was suggested that further improvement could be made on the task by working to create a bigger and more comprehensive dataset to facilitate the training of deeper network structures and to allow further insight into how domain professionals discriminate between images.

Keywords: Image manipulation, CNN, LSTM, image classification, time series classification

1 Introduction

The present paper extends upon work presented in Mills-Thom [6] 2020, which attempted to classify digital images as manipulated or unmanipulated based on aggregate eye gaze data using a simple feed forward neural network (NN). While this paper focused on explaining the network decisions, the present paper focusses on improving classification results on this problem by utilising DL techniques on the full eye gaze sequence dataset.

1.1 Previous Work

This paper follows up on previous work in the same domain. Mills-Thom [6] 2020 used a shallow feed-forward neural network on an augmented set of this data to predict image manipulation status from aggregate eye gaze data. This experiment yielded around 64% accuracy, which although low, was an improvement on the original human performance, which was 56% [3]. Detection of image manipulation is covered in digital forensics literature, employing convolutional neural networks or statistical methods for determining if an image has been manipulated [1]; however, predicting image manipulation from eye gaze data appears to be novel topic.

General image manipulations can range from simple scaling or addition of noise, to full re-composition and seamless combination of multiple images. These manipulations can range from obvious to invisible, making the task extremely hard to generalise. From this perspective, utilising the general intelligence of the human brain could be a good approach for solving this classification problem.

While there is little work in this specific domain, the general domain of time series classification using DL has become a topic of interest in recent years [4]. Time series classifiers either learn using hand engineered or use end-to-end deep learning models which incorporates feature learning into the training process. Deep neural networks have recently shown good results when applied to time series classification tasks and do not rely on hand engineered features [4].

1.2 The Dataset

The dataset used in this paper comes from Caldwell, Gedeon, Jones and Copeland's [3] 2015 experiment on human identification of digitally manipulated images. The experiment presented participants with an image that was either an original image or a digitally manipulated version of that image, the participants freely viewed the image before indicating whether they thought the image was manipulated, not manipulated, or if they weren't sure. The researchers used eye tracking to record the participants gaze over the image and recoded data detailing fixations on the image. These data points included the location of fixation, how many gaze points were observed in that area, how long the fixation lasted, and some unique identifiers for the participant, image, and specific fixation. Data for a participant viewing an image is formatted as a sequence of these data points, with the length of the sequence varying between each participant-image pairing. The researchers also recorded which areas of each image had been manipulated using four (x, y) points to create a bounding box around the area.

The dataset used in this paper is a subset of the original Caldwell et al dataset. It only has data from the last five images in the image set (depicted in Figure 1) and not all 80 participants viewed all 5 images, resulting in 372 sequences instead of the full 400. A sample data point is shown in Table 1 (over page).

Table 1. A sample data point from the Caldwell eye gaze dataset

Fixation ID	Participant ID	Image ID	X Pos	Y Pos	Start time	Stop time	Duration	Samples in fixation
39300	1	10	498	611	968.172	968.239	0.067	5

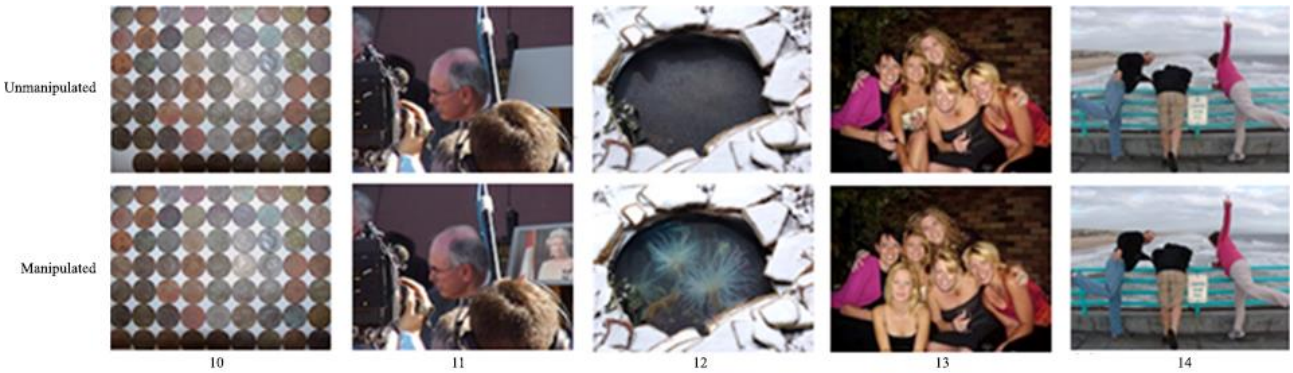

Figure 1. The manipulated and unmanipulated versions of the five images participants were shown [3]

1.3 Exploratory Data Analysis

Since both humans and shallow learning found it difficult to separate manipulated from untouched images, it is worth manually investigating the structure of the data to see if the two categories are theoretically separable, and how the data can best be used to facilitate learning.

The data shows a difference between average time spent viewing the manipulated area when the image has been manipulated versus when it hasn't, indicating manipulated images are of increased interest to us even if we don't realise it. This suggests that while we may not know it, our gaze over the image is different when the image has been manipulated, and a machine learning approach may be able to identify this difference based on our gaze patterns. However, this effect is mediated by the image being viewed, with images 10 and 14 showing little difference between groups. Images 11 through 13 on the other hand show obvious differences between groups. This effect is due to images 10 and 14 having objectively more seamless manipulations.

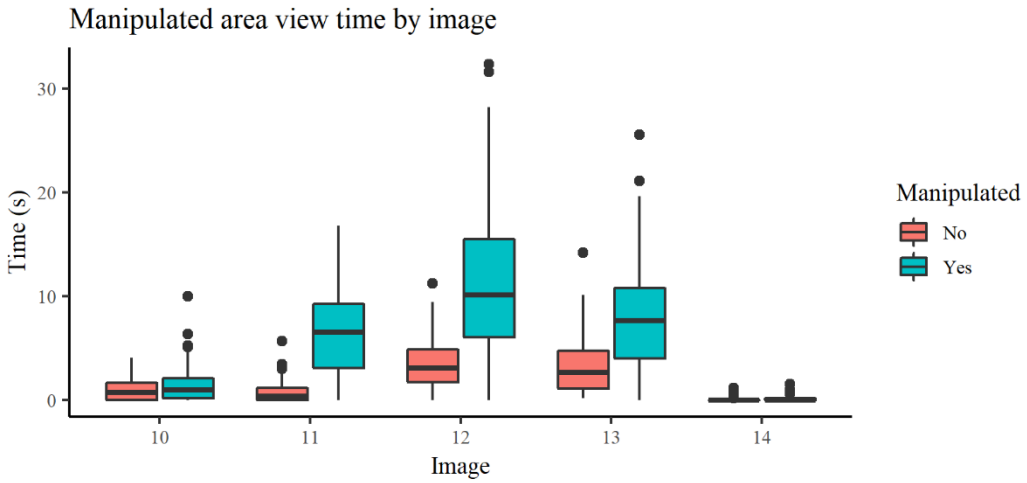

Figure 2. Distribution of time spent looking at manipulated area by image and manipulation status

This result was echoed by the average human accuracy results for each image, which were 51%, 67%, 57%, 64%, and 52% for images 10 through 14 respectively. The difference between view times in image 12 and its poor classification accuracy seem to suggest opposite conclusions, however as seen in Figure 1, the manipulated area is large and central, thus boosting this statistic somewhat.

Learning approaches on this dataset may be impacted by these effects, it would be expected that a model trained and tested on images 11-13 would yield better results than if 10 and 14 were included. While this wouldn't constitute a complete solution to the classification problem, it is an idea of interest to explore.

The class labels are evenly distributed and there is no significant image bias in the data, although image 14 may be slightly underrepresented. A simple random selection approach for the train and test sets will function well under these conditions.

1.4 Candidate Architectures

Given the dataset is sequential in nature, with the length of each sequence varying, DL models that can understand the relationship between data points over time and that can handle variable length input are of specific interest.

LSTM cell-based models have traditionally been used to classify sequences of this nature, particularly textual data for language translation [8], text generation [2], and sentiment analysis [7]. The latter of these examples demonstrates that LSTMs can be applied to sequences of varying length to classify the sequence with a label, which is basically our classification problem.

CNNs have also found success when applied to similar problems [4, 8, 9]. CNNs can utilise filters to identify features over set lengths of the sequence, which allows them to encode time-based information, or context, into the feature maps.

Ensemble methods utilising both CNNs and LSTMs have also proven successful in time series classification [5] and seem particularly interesting for this problem. These methods use CNN layers to learn features of the input and encode a feature response vector over time, which is both resilient to noise and able to encode context. The output is then fed into an LSTM which can encode contextual information deep into the sequence. These attributes are necessary for the current classification problem, but a relatively complex architecture may struggle with the small dataset available.

2 Method

LSTM models, CNN models, and a combination of the two were tested, using 50 to 100 epochs of training and taking the maximum stable test accuracy as the statistic of comparison. Initial tests were done using a single 80/20 train-test split, if the model appeared to perform well, it was evaluated using 5-fold cross validation using the average final test accuracy across the 5 folds as the accuracy statistic for the model. Stochastic gradient descent was used to train each model and binary cross entropy (BCE) loss was used for evaluation. The training loss, training accuracy, and test accuracy were recorded and informed overall performance of the model.

2.1 Pre-processing

For every model, the data was pre-processed in the same way. A column was added encoding whether the gaze point was in the manipulated area or not; the participant and fixation IDs were dropped as they encode no general information; the start and stop time columns were dropped as the duration column supersedes them; the positional and duration columns were min-max scaled by image; and the image IDs were one hot encoded.

The positional data was kept (and scaled by image) because the general path of image viewing could reveal something about the nature of the image, and the DL techniques to be used in this paper will be able to represent relationships between successive data points. It was scaled by image to control for potential differences in image size between images; the relative position of gaze events is only useful when viewed in the context of the image being viewed.

Min-max scaling in general was used to control the implicit weightings of inputs. Brining all inputs into the range 0-1 guarantees equal initial importance and makes it easier for the model to identify extreme values. Since this is a novel domain, there is no reason to assume any attribute has more importance than any other, hence brining all data into the same range is objectively the best initialisation.

The result of this processing is a 372x*S*x10 dataset, where *S* refers to a variable sequence length. Since *S* varies, the data cannot form a contiguous tensor, and thus batch learning is not possible without padding the data which was not done in this experiment. A single sequence in the dataset has shape *S*x10; *S* data points consisting of 10 attributes.

When using a CNN layer first, this sequence is transposed to have shape 10x*S*; where each row contains a single attribute over the sequence.

2.2 Model Selection

Three basic architectures were investigated: CNN based models, LSTM based models, and an ensemble model consisting of a CNN and an LSTM. The final layer in every model was a fully connected neural network using a Sigmoid activation function to facilitate binary classification and the use of BCE loss. The specific architectures of these models varied. For example, the number of convolutional layers was varied, dropout layers were sometimes used, the number of hidden layers in the final fully connected layer were varied etc.

2.3 Hyperparameter Selection

For this experiment, hyperparameters refer to parameters that do not change the structure of the network. This includes filter width for convolutional layers, filter width for pooling layers, size of the hidden state vectors in an LSTM layer, the learning rate for the algorithm, the number of feature vectors in a convolutional layer, the amount of epochs to train for, and the probability in a dropout layer.

2.4 Testing Method

Hyperparameter testing was briefly done for each model unless the model showed increased performance over previously tested models, in which case an attempt to fine tune the hyperparameters was done. Oftentimes, model testing and hyperparameter testing were done in parallel, where model and parameter changes were made at the same time.

The testing was not exhaustive and was instead guided by results and general understanding of what approaches should work well on the given data. To gauge understanding on the effects of different configurations, three general values (or architecture decisions) were investigated: a small value or size, a medium value or size, and a large value or size.

3 Results and Discussion

For the Caldwell [3] dataset, the architecture that found the most success was an ensemble method, combining a single convolutional layer and a moderately sized LSTM, discussed in detail in section 3.1. It was found that CNN models were incapable of exceeding 55-60% test accuracy, regardless of size and complexity. However, these models did appear to learn the training data quite quickly, indicating that was capable of learning some of the structure in the data, but that structure wasn't well-generalisable.

LSTM models were found to be slightly more reliable, achieving accuracy in the 55-60% range consistently. It was found that medium sized hidden states (between 8-16) performed the best, although both large and small networks didn't perform poorly in comparison to the CNN results. LSTM models took notably longer than CNN models, making parameter tuning highly time consuming and difficult.

Since the ensemble method incorporates an LSTM, it also suffers from long training time. However, it was found to produce better and more reliable results. It was found that a relatively small architecture in each layer was able to generalise the training data the best. Less than 5 filters of width 10-15 using only one convolutional layer at the start seemed to produce good feature vectors; the following LSTM layer with hidden states of size 10-16 seemed to learn context well; and a feed forward NN layer with no hidden layers produced the best accuracy results. These observations may be partly due to the size of the dataset – a bigger dataset may facilitate the training of more complex models that may reach higher levels of accuracy on this task.

3.1 Discussion of Best Performing Architecture

The best performing model was an ensemble method, utilising a single convolutional layer to learn three filters and produce three feature vectors. This layer uses a filter width of 10 and utilises zero padding of length 5 on either side of the input vectors to maintain the same length in the output vectors. The resultant feature vectors are then run through a ReLU activation function before going through a max pooling layer using a filter width of 3. The three feature vectors are then transposed to give a sequence of length 3 vectors which contain the feature response for each time step in the input, minus the values 'lost' by the max pooling operation. This sequence is then fed into an LSTM using hidden states of length 12, which output a vector of the same length for every step of the sequence. The last output of the LSTM is taken and has dropout applied to it with a probability of 10% and is then fed through a fully connected layer with one output neuron which uses a Sigmoid activation function. This model was trained using stochastic gradient descent with a learning rate of 0.01 and used the BCE loss function.

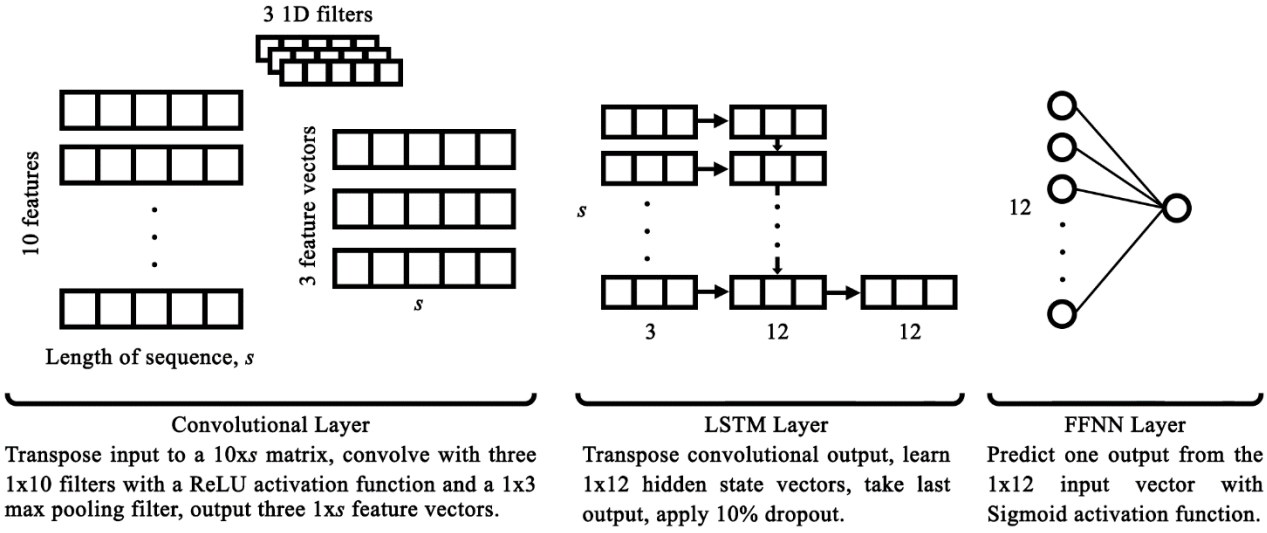

Figure 3. The specific architecture for the best performing method

This model was able to achieve 67.82% test accuracy on the modified Caldwell [3] dataset. This statistic comes from the average accuracy from the last 10 epochs from each of the 5 folds, making it resistant to noise. The average accuracy graph shown in Figure 4 displays a slight increasing trend remains at the end of the graph, which indicates this may be a pessimistic value. However, given the training sets only contain 74 sequences, and the accuracy trend shows relatively high variance, this statistic is probably representative of the model.

This test accuracy is higher than both the human participants in Caldwell et al's study (56%) [3], and the simple neural network presented in the predecessor to the current paper (64%) [6]. The observations made in section 1.3 and the increased performance from this sequence of studies shows that while people may be bad at detecting manipulations in an image, their visual processing of the image can be an indicator of manipulations. This suggests that humans have an innate understanding of images but are unable to consolidate this understanding into an accurate decision or observation.

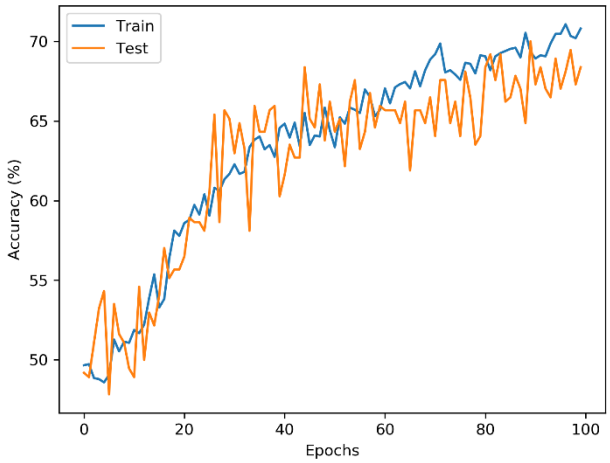

Figure 4. Train and test accuracy per epoch averaged over the 5 validation folds

3.2 Simplifying the Problem

As identified in section 1.3, the manipulations in images 10 and 14 were harder to detect than in the others, evidenced by the difference in detection accuracy between images. Image 12 also had lower accuracy, but it showed a significant change in the total viewing time of the manipulated area, whereas images 10 and 14 did not; implying that a difference can be found between the two classifications. As such, the model described in the previous section was applied to a further subset of the Caldwell dataset, where only data for images 11, 12, and 13 were kept. This reduced the dataset to only 225 sequences, reducing the size of the dataset by approximately 40%. It was expected that the model would achieve better results on this data, as the data from images 10 and 14 could be simply considered noise.

When trained and tested on this data, the model achieved 72.53% accuracy; a 5% improvement on the performance on full dataset despite having 40% less data to train with. This also represents a 10% increase on the average human classification accuracy for these three images (62.72%). This result does however carry the caveat that the test sets for 5-fold cross validation on the reduced dataset contain only 45 sequences, meaning the results may not be robust.

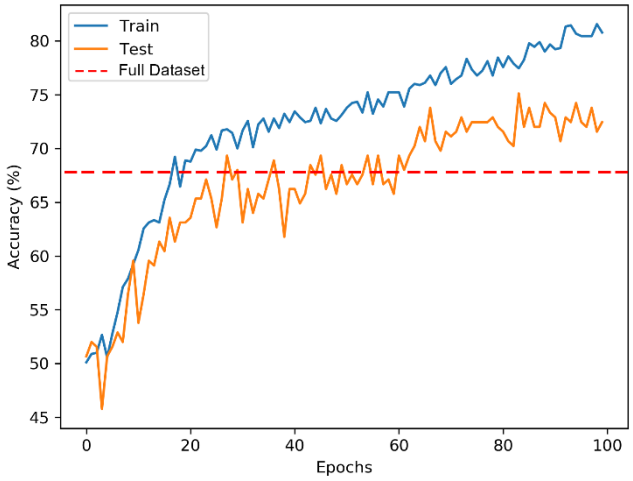

Figure 5. Train and test accuracy per epoch averaged over the 5 validation folds on the reduced dataset, red line shows the test accuracy of the same model on the full dataset

The test accuracy shown in Figure 5 has much less variance than the test accuracy for the full dataset. This suggests that images 10 and 14 do indeed add more noise to the data than discriminative ability. It also shows higher train accuracy over the same training time using the same architecture, again providing evidence that images 10 and 14 are hindering the ability of the models.

While training and evaluating on the 'easy' data undermines the complexity of the broader task, the insights from successful training on a simpler dataset could be useful for building up an understanding of how models solve these problems and what factors help or hinder them.

3.3 Results Summary

All networks outperformed humans, with the previously discussed ensemble architecture achieving the best results. The testing results are summarized in Table 2 below:

Table 2. Average range of accuracy results for tested architectures with varied parameters

Model	Accuracy Range	Varied Parameters
Human	56%	N/A
Simple NN	~64%	No. hidden neurons, no. hidden layers, activation function
CNN	53-58%	No. filters, filter width, pooling width, activation function, FFNN params
LSTM	55-60%	Hidden state size, FCNN params
Ensemble	64-68%	CNN params, LSTM params, FCNN params
Ensemble on reduced dataset	~72%	Used best performing ensemble architecture

4 Conclusion

While the classification problem is far from solved, applying DL techniques to the original sequence dataset progressed results once more. An ensemble method combining a CNN with and LSTM was able to outperform both the feed forward network discussed in the previous paper [6], and the human participants of the original study. The results this paper presents imply that *how* we observe images carries more discriminative power for identifying manipulations than the overall result of our observations; and both can be better utilised by machine learning techniques than by our own brains.

Improvement aside, 68-72% is still very low for a binary classification task, and there is still room for improvement. Future work should look to obtain a more extensive, and comprehensive dataset. More data will allow for the training of deeper networks, allowing for more complex understandings of the data, which may be necessary for improvement considering even humans show poor accuracy. “More comprehensive” could be defined and achieved in many different ways; a dataset with similar manipulations of similar classification difficulty could be useful, and value could also be found in a bigger, heterogenous dataset as well. Data from participants familiar with digital manipulations such as digital artists or designers could give better insight into how professionals in the domain classify images. Finally, the data collected could be expanded to include brain activity as well, such as electroencephalography (EEG) data or functional magnetic resonance imaging (fMRI) data to further understand how the brain processes these images.

5 References

[1] Bayram S, Avcibas I, Sankur B, Memon N (2006) Image manipulation detection. Journal of Electronic Imaging 15(4).

[2] Boran M (2016) Web Log: World first as AI writes sci-fi movie 'Sunspring'. The Irish Times. Available: https://www.irishtimes.com/business/technology/web-log-world-first-as-ai-writes-sci-fi-movie-sunspring-1.2685475

[3] Caldwell S, Gedeon T, Jones R, Copeland L (2015) Imperfect understandings: a grounded theory and eye gaze investigation of human perceptions of manipulated and unmanipulated digital images. Proceedings of the 1st World Congress on Electrical Engineering and Computer Science Systems and Science, Barcelona.

[4] Fawaz H, Forestier G, Weber J, Idoumghar L, Muller P (2019) Deep learning for time series classification: a review. Data Mining and Knowledge Discovery 33:917-963.

[5] Karim F, Majumdar S, Darabi H, Chen S (2017) LSTM fully convolutional networks for time series classification. IEEE Access 6:1662-1669.

[6] Mills-Thom J (2020) Extracting rules for the identification of manipulated versus unmanipulated digital images. Unpublished.

[7] Nasukawa T, Yi J (2003) Sentiment analysis: Capturing favorability using natural language processing. Proceedings of the 2nd international conference on Knowledge Capture, Florida 70-77.

[8] Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. Neural information processing systems, 3104-3112.

[9] Zhao B, Lu H, Chen S, Liu J, Wu D (2017) Convolutional neural networks for time series classification. Journal of Systems Engineering and Electronics 28(1):162-169.