Detecting Confusion Using an LSTM

Audibert Leo
Simon Fraser University
Burnaby, Canada
laudiber@sfu.ca

Barer Simon
Simon Fraser University
Burnaby, Canada
sbarer@sfu.ca

ABSTRACT

In a retail setting, it is crucial for employees to quickly and effectively identify which customers require assistance, and which do not. Unfortunately, due to numerous reasons, this is not always possible. In this paper, we have undertaken development of a neural network capable of identifying confusion in customers in real time. We utilized multiple different model variations in order to achieve the best possible results. The advantage of this approach is a learning platform that will get better the more data it has to learn from. Our work will allow retail employees to easily intercept the customers who need help the most, maximizing efficiency and customer support.

KEYWORDS

datasets, neural networks, confusion detection, long short-term memory network

1 INTRODUCTION

Retail environments may vary in size, service, and number of employees, but there is always a need to assist customers with their purchases. Unfortunately, it is not always feasible to see every customer at once, and even more difficult to identify if they truly need help or are just happy browsing on their own. Studies have shown that customer service plays a crucial role in customer satisfaction, loyalty, and brand reputation [2]. In fact, the likelihood of a customer being displeased with a company is strongly correlated to the company's existing reputation [4]. This positive feedback loop is crucial to brand success, and this is why companies such as Apple have placed such emphasis on support [1, 3]. In fact, it is not just the quality of service that has been found to be important, but also the timeliness in which the support is provided [8]. Therefore, the ability of business to identify which customers need help, and to identify them as quickly is possible, is paramount to not only the short-term success of that business (ie. making a sale), but also the long-term satisfaction and positive awareness and reputation of the brand.

Businesses have tried to address this problem in multiple ways. Walmart famously installs greeters at all entrances to their premises, in the hopes of quickly aiding shoppers upon entry. However, this solution does not provide aid to those already in the store. Everyone has had the experience where they have a harder time tracking down an employee than finding the product they were looking for to begin with. Retail stores have tried to reduce confusion through organization of products, large signage at the entrances to each aisle, and more. Unfortunately, with so much competition in the retail space, and every brand vying for attention on the shelf, modern advertisements and packaging can often be overwhelming, misleading, or unclear, only contributing more to customer confusion [9].

Other solutions have included in-store downloadable phone apps, with call buttons to request help to your location. However, this solution requires the customers downloading an app, opening their phones to data-mining and security risks, which in today's climate, most are unwilling to do.

Adding to this problem is the issue of identifying confusion itself. Many customers enter a store knowing exactly what they've come to purchase, and don't require any assistance whatsoever. Worse still, some customers don't want to be bothered by employees, but to just mind their own business and accomplish whatever task they've set out to accomplish. These customers in particular would likely chaste at being interrupted by an employee offering assistance, thus achieving the opposite of the intended effect. In both of these cases, employee involvement would result in a net loss, as not only have they interrupted a customer who did not desire help, but the employee could have potentially aided another customer who did need help instead. Unfortunately, confusion is a complex emotion [5, 6], and is not always easy to identify. To compound this issue, while companies focus on training staff in their products, how to highlight certain facts, and what language to use, training employees to identify confusion is often overlooked entirely.

It is evident that a new tact is needed in order to solve this problem. Neural networks provide the ability to aggregate large amounts of data to solve problems such as this. By analyzing a customer's movements and posture, it should be possible to identify their level of confusion in real-time. Once positive identification of a subject is made, an employee can immediately be alerted, where they can then intercept the confused customer and bring about a positive support experience.

2 APPROACH

2.1 Dataset

Our dataset was collected by Zhitian Zhang et al., and was kindly provided to us to use in our research. The set of experiments was conducted in the computing science robotics lab in Simon Fraser University, Burnaby, BC, Canada, and its aim was to gather data containing common gestures, movements, and other behaviours that may indicate humans' navigational intent that may be applicable to robots. The experiment simulated a shopping scenario where human participants came in to pick up items from his/her shopping list and interact with a Pepper robot that was programmed to help the human participant. Visual data, in the form of motion capture from 5 cameras, was collected. The datastreams of all cameras were fed across a network to a single computer, which then recorded and saved the data. A total of 108 people participated in this study, resulting in 540 distinct data samples.

The resulting data was then converted into MPEG format for ease of annotation. Each experiment in the dataset was filmed from

at least 4 vantage points. After extracting the frames from every video, they were run through OpenPose, utilizing the COCO model. OpenPose is able to identify the coordinates of the subject's body and limb, and overlay a wireframe on the extracted frames. The resulting data points were saved to a JSON file. We were unable to make OpenPose run on GPU with CUDA, resulting in around 400 hours of processing time distributed over two machines for two weeks. In addition, we constructed a custom annotation tool that allowed the videos to be annotated with a Confusion Score on a frame by frame basis. These scores ranged from 1 - "not confused" - to 3 - "confused". The annotation was performed exclusively by the authors of this report, and an effort was made to ensure consistency between annotators.

There were numerous hurdles and obstacles related to the dataset and its annotation, which we have described at length in our Discussion.

The openPose coordinates and annotated Confusion Scores were then correlated with one another. These data points represent our X and Y variables for the neural network models, respectively.

For every frame of every video, an aggregation script stitched the OpenPose data to the labels. Frames with the wrong individual or no individual were removed in preprocessing. Due to OpenPose detecting only one individual at a time, it resulted in a bystander sometimes being processed. The centroid of the OpenPose points of a frame were calculated. If the centroid of the next frame was a greater distance from the centroid of the previous frame, a different individual was assumed to be detected and it was removed. The maximum allowable distance between two centroids in two successive frames was determined experimentally.

A limitation of the OpenPose data points is that they are coordinates on the 2 dimensional image plane. We were unsure as to whether different cameras were going to generalize well.

2.2 Data Preprocessing

Our data was then preprocessed through the use of centroid detection with a maximally allowed variance. In essence, this stripped away any frames where the subject identified by openPose was not the test subject, but instead either the Pepper robot, or one of the staff on hand for the experiment. If four or less faulty frames separated a sequence with the same individuals, it was stitched back together without the faulty frames. If more than four faulty frames were found, the two sequences were saved as separate sequences. Four frames was chosen, as this was close to the average frame rate of the videos, meaning that the maximum allowance of dropped frames roughly equalled one second. A minimum sequence length of 40 frames was chosen, in order to provide enough context to our models for a single sequence, regardless of what window size we chose for the models (more on this later). All OpenPose data points were normalized so as to have zero mean and unit variance. In this way, we stripped away any frame that had either no subject present, or had the wrong subject analyzed by openPose. Additional processing included removing any frames that did not have 54 openPose features, and removing any frames with a Confusion Score of 0 (meaning that there was no test subject present, or that openPose misidentified the subject), as well as downcasting the Scores from 1-3 to 0-2 in order to obtain better results from our models.

2.3 Models

Our models were constructed using the Keras library in TensorFlow. The experiment consisted of three distinct models: a simple Sequential Neural Network (SNN), a single layer Long Short Term (LSTM) Neural Network, and multi-layer LSTM. The SNN was chosen to function as our baseline to evaluate the efficacy of the other two models. Long short-term memory neural networks are a type of convolutional network. Their inner memory makes them perfectly designed for sequential data [7].

3 EXPERIMENTS AND RESULTS

Before preprocessing, our dataset contained 146 annotated videos, composed of 42844 distinct frames. After preprocessing, our data contained 24415 frames in 201 normalized sequences. Of these frames, 9463 had a Confusion Score of 1 ("Not Confused"), 6817 had a Score of 2 ("Maybe Confused"), and 8135 had a Score of 3 ("Confused"). The remaining 18469 frames were removed because they had a zero Confusion Score (3955 frames), had invalid centroid positions, or were part of a sequence below the minimum sequence length. Of the 201 normalized sequences, the mean length was 121 frames with a standard deviation of 102, a maximum of 683 and a minimum of 40. After our Confusion Scores were reduced to a range of 0-2, the mean was 0.945 with a standard deviation of 0.847.

For the baseline Sequential Neural Network (SNN), the dataset was split into 3 distinct, non-overlapping sets: training (64%, or 15625 frames), validation (16%, or 3907 frames), and test sets (20%, or 4883 frames). These percentages were chosen in order to ensure that all models had a large enough pool of training data. The model three Dense layers were used, with decreasing numbers of nodes, from 50 to 10 to 3, with the first two layers utilizing "relu" activation, and the last layer utilizing "softmax" for probabilistic prediction of the Confusion Score. The SNN was run for only 10 epochs, as no appreciable improvements were observed with more, as well as to prevent overfitting. Cross-validation was performed at the end of each epoch.

For the LSTM models, the dataset had to be reshaped to include a temporal component, "window size", with the resulting shape being [number_of_windows][window_size][54 features]. Window sizes of three magnitudes - 5, 10, and 20 - were chosen, in order to observe what effect temporal context might have on the LSTM results. Two LSTM models were built, a simple LSTM (sLSTM) consisting of a single LSTM layer, and a complex LSTM (cLSTM), comprising three LSTM layers with Dropout layers of 0.2 following each layer. Both models concluded with a Dense layer with "softmax" activation in order to output probabilistic predictions of Confusion Scores. The number of hidden nodes chosen for all LSTM layers was determined using the formula Nh = Ns / (alpha * (Ni + No)), which equalled 9 nodes for 5 window size, 4 for 10 window size, and 2 for 20 window size. An early stopping callback function was used to prevent overfitting, which monitored value loss. Cross-validation was performed at the end of each epoch.

In general, it was found that more training epochs led to a higher accuracy rate on the training set (as one would expect), but only marginal improvements, if any, on the validation set (see the Appendix for graphs and tables depicting the accuracy and loss of each model per epoch). Loss was also measured per epoch, and it

was found that it generally decreased with more training of the model, with the exception of the Simple-5 LSTM model, which varied widely, forming no discernable pattern across the 6 epochs.

On training and validation sets, it was found that there was little statistically relevant difference between the complex and simple LSTMS, with both producing accurate predictions roughly 40% of the time. It was found that window size did affect the predictive power of the models, with roughly 37% correctness at 20 vs 40% at 5 and 10. However, it is possible that this difference may be due to the amount of data available to train the models, and that increasing the window size effectively decreased the number of data points overall. None of the 6 LSTMs produced highly confident predictions, with no model being more confident than 71% or less confident than 12%. Contrast this with the SNN, which had confidence rates as high as 99% and as low as 0.7%. (See the Appendix for a table of predictions for all models of a given subset of the test set)

It was found that the SNN had the highest rate of accuracy on the validation set, as well as the most and least confident predictions. However, when it came to the test set, the SNN had the most confident right answers, as well as the most confidence in answers that were incorrect. The SNN had the lowest correct prediction percentage, at 47.266%. The sLSTM had a higher correct percentage than the cLSTM at only a window size of 5, with the cLSTM being more effective than its counterpart at 10, 20, and 40 window size. The most effective model was the cLSTM with 20 window size, at 70.676% accuracy (where 33% reflects "random chance"). However, it is noteworthy that for all LSTM models, the mean confidence with which it predicted correct answers was only 2-7% above random chance, and the incorrect answers 3-19% above random chance. When the LSTMs did predict incorrectly, the correct answer had a mean prediction of 25-31%, which ranged 5-16% less than the answer the model did predict. The 40 window size was included to confirm that there is a maxima in training around 20, and that the accuracy did not keep increasing with frame size.

4 DISCUSSION

The main source of frustration with this project was the use and annotation of the dataset. We selected this topic in large part because we would be provided with a completed dataset, and believed that we would therefore be able to allocate more of our resources and time into developing better models. Because of the nature of our data - 5 cameras capturing the same test - we initially constructed our annotator to annotate all 5 feeds simultaneously. This had numerous benefits, as beyond saving time, it ensured annotation consistency between video files, something that couldn't have been accomplished otherwise. Upon successful development of our annotator tool, we realized that the camera from the Pepper robot was a different frame rate than the other four cameras, preventing its usage in our tool. Upon further investigation, we found numerous intractable issues with the dataset that we were unable to overcome. Chief among these issues was the fact that the cameras did not record the video locally, but instead transmitted them over a network to a computer for recording. The consequence of which was the misalignment in both temporality and data frames of the camera feeds. Because of this, it wasn't technologically possible to align the videos for simultaneous annotation. This meant that we would be faced with either doing 5 times the amount of work to annotate the entire dataset, or expend the same amount of effort to obtain a dataset 1/5th the size we had anticipated. Additionally, this also resulted in the inability to utilize the 3D points tracked from the participants' helmets, thus further reducing the amount of data available to us. As we already had concerns regarding whether we would have enough data to adequately train our models, we were left with no choice but to spend significantly more time annotating the dataset, on top of the time we had spent determining the source of our issues, and attempts to mitigate these issues.

Further complications with data resulted from the design of the test itself. The purpose of the models developed in this project is to aid retail employees in identifying confused customers so that they can be provided assistance. Working backward logically from this goal, it stands to reason that the AI must first identify the customer, then notify the employee, and then have the employee approach the customer. This whole process takes considerable time, and as such, the confusion being identified in the subject can't be fleeting. It is of no value to any party involved if the AI identifies a confused customer, but by the time the employee is notified and then approaches the customer, they are no longer confused. Following this logic, it becomes clear that in order to properly train our models to achieve the desired result, the dataset should contain instances of authentic, uninterrupted, long-lasting confusion (on the scale of at least 30 seconds or so). Unfortunately, due to the nature of the experiment's construction, this situation almost never occurred. Typically the subject would walk from table to table, checking their list against the 8 items on each table while staying relatively motionless in front of the table, and then either grabbing an item present on their list, or moving on to the next table. As a result, the only confusion observed in the majority of tests was in the brief moments while the subject scanned the 8 items on a given table while cross-referencing their list. This is not the type of confusion that is actionable in our use case for this model.

With these goals in mind, it is evident that an experiment should have been constructed that would induce authentic, prolonged confusion. As a prototypical example of the type of confusion we should be trying to synthesize, envision a customer in a hardware store, standing in front of a shelf of paint cans for minutes on end, trying to ascertain which can is right for their specific needs. The customer's behaviours in this scenario would likely include looking at the front labels, reading the description and instructions, determining which size of can would provide the correct amount of paint, which brand is best, etc. If the customer knew exactly what specific can of paint they wanted when they entered the store, the only confusion they would experience is in the brief time it would take them to scan the shelves for the item, which would not be authentic or prolonged. Unfortunately, the latter is what the performed experiment typically presented.

Therefore, a better constructed experiment would have proceeded as follows: Instead of 6 tables spaced apart, have a single center aisle consisting of two shelving unit rows with a barrier in between, with 3-5 levels of products within each shelf. To simplify variables, and to ensure that a subject does not merely skip over an item they cannot find and wait until the end to ask an agent, discretize the test to one item per trial. Do not provide the participants with a specific list, and instead of always giving them

an exact request, sometimes provide them with overly vague requirements for what they are expected to obtain. As an example of such an instruction, one could tell the participant: “The item I want is blue, although it might be green. I don’t think it’s that heavy, and I’m pretty sure it doesn’t cost that much.” In this scenario, the shelves would contain multiple blue objects, but also green objects (or teal, turquoise, etc.) of varying weights and prices. Because of the vagueness of the requirements, the participant will be in a much greater state of confusion, and will be looking up and down aisles and shelves, perhaps grabbing two similar products in each hand and thinking about which one’s weight matches the request. These are signs of prolonged and authentic confusion, and in a real world scenario, are the types of issues that would lead customers to seek the aid of an employee. Given a ratio of, say, one in every four requests being confusing, with the other three being more explicit, a strong baseline for what “confused” looks like, versus “not confused” will be established. Additionally, assuming that the annotators are unaware of which trials were explicit instruction or confusing instruction, the annotations are guaranteed to be blind and unbiased, thus ensuring a reliable and valid dataset.

As a result of the dataset used, and the difficulty with which even the annotators had in discerning if someone is confused or not, it is not surprising that our models did not produce predictions with a high confidence. While all models has a success rate above random chance, no model was reliable enough to be applicable to the use case this project was designed to address. The most successful model by percentages, Complex-20, only seems right because it mostly predicted all answers as "Not Confused", leading to a high percentage of correct answers for that Score only, but only getting 100/2772 correct for the other two Confusion Scores.

It is our expectation that with a greater pool of training and validation data, as well as training utilizing k-fold, the models could both improve their correctness, as well as their confidence, and that larger window sizes may actually become more effective, if there is enough data to support them. That the confidence with which our models predicted answers correctly, 36-41%, and that all predictions ranged between 10-71% confidence, also reflects how similar a confused or non-confused person appears, based off of body language alone. It was difficult to identify authentic confusion in the dataset when annotated by humans, so it stands to reason that an AI trained on this data would also not be confident in its answers.

We were surprised with how few epochs were required to train our models, and by how little each epoch seemed to improve the accuracy and loss values. We suspect that this would also not be the case if we had a larger dataset with which to work with, and/or used k-fold for splitting up the train and validation sets. It was interesting to note that the SNN had the highest and lowest confidence scores, and yet predicted the lowest percentage of correct answers of any of our models.

Future improvements to this work would include creating a new, more robust dataset with which to train the model, building the model using k-fold validation and frame flipping to increase the efficacy and volume of the data during training, and build the model to respond off of live events in real time, in order to provide the full vision of the use case that which we built the model to solve. Due to our models inability to correctly predict "Maybe Confused", it would also be interesting to further consolidate our 3-way prediction model into a binary “confused”/”not confused”, and investigate whether that increases the correctness and confidence of the models. Once the model produces reliably accurate results, it would be interesting to try and train it on datasets of more than a single person shopping at a time.

5 CONCLUSION

Our project set out to annotate a dataset of retail shoppers based on their confusion level, and build a neural network capable of identifying confusion with a high rate of confidence and accuracy, in real time. It was found that, while our data was not ideal for these purposes, our best model produced the correct answer 71% of the time, over twice that of random chance. With more data and time to improve the model, we believe that this project could provide real and tangible results to retail companies, and their customers.

ACKNOWLEDGMENTS

To the SFU Robotics Lab for kindly providing us with the raw dataset.

REFERENCES

[1] Steve Denning. 2012. Another Myth Bites The Dust: How Apple Listens To Its Customers. https://www.forbes.com/sites/stevedenning/2011/08/26/another-myth-bites-the-dust-how-apple-listens-to-its-customers/#41832d41cde5

[2] Craig C Julian and Bashar Ramaseshan. 1994. The Role of Customer-contact Personnel in the Marketing of a Retail Bank s Services. *International Journal of Retail & Distribution Management* (1994).

[3] Ms Nandhini Rajasekaran and MN Dinesh. 2018. How Net Promoter Score Relates To Organizational Growth. *Int. J. Creat. Res. Thoughts* 6, 2 (2018), 2320–2882.

[4] Anne L Roggeveen, Neeraj Bharadwaj, and Wayne D Hoyer. 2007. How call center location impacts expectations of service from reputable versus lesser known firms. *Journal of Retailing* 83, 4 (2007), 403–410.

[5] Paul Rozin and Adam B Cohen. 2003. High frequency of facial expressions corresponding to confusion, concentration, and worry in an analysis of naturally occurring facial expressions of Americans. *Emotion* 3, 1 (2003), 68.

[6] Paul J Silvia. 2009. Looking past pleasure: anger, confusion, disgust, pride, surprise, and other unusual aesthetic emotions. *Psychology of Aesthetics, Creativity, and the Arts* 3, 1 (2009), 48.

[7] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[8] Gail Tom and Scott Lucey. 1995. Waiting time delays and customer satisfaction in supermarkets. *Journal of Services Marketing* (1995).

[9] Gianfranco Walsh and Vincent-Wayne Mitchell. 2010. The effect of consumer confusion proneness on word of mouth, trust, and customer satisfaction. *European Journal of Marketing* (2010).

A FIGURES

Model	# of Epochs	Loss	Accuracy	Val_Loss	Val_Accuracy
SNN	10	1.066891	0.420416	1.070016	0.414384
Simple-5	6	1.070468	0.417093	1.082371	0.397954
Complex-5	6	1.079145	0.406658	1.081864	0.413811
Simple-10	8	1.083720	0.408008	1.084431	0.419437
Complex-10	10	1.083634	0.400448	1.074926	0.425575
Simple-20	6	1.084620	0.379295	1.085297	0.381633
Complex-20	10	1.085828	0.372692	1.084120	0.429082

Figure 1: Results at each epoch for all models

Model	Max Probabilities Per Confusion Score			Min Probabilities Per Confusion Score		
	0	1	2	0	1	2
SNN	0.9860669	0.9906786	0.9156548	0.007799558	0.0036209302	0.0013696807
Simple-5	0.71494097	0.4628453	0.47574145	0.2387822	0.12013867	0.15189834
Complex-5	0.5911271	0.45438337	0.48040712	0.23570153	0.14900316	0.18297432
Simple-10	0.4942418	0.3579002	0.4308607	0.24106814	0.19486856	0.27841753
Complex-10	0.49779302	0.34884802	0.44902596	0.29660764	0.25188974	0.15368256
Simple-20	0.61199534	0.4020795	0.35933423	0.25672084	0.10662991	0.28133103
Complex-20	0.5317115	0.35338846	0.35167396	0.3007093	0.14419152	0.324096

Figure 2: Max and min probabilities reported in test set predictions

	Correct	Mean Correct	Incorrect	Mean Actual Answer	Mean Incorrect Answer	% Correct
SNN	1644	0.45783	3239	0.27643	0.45531	33.668%
Simple-5	1814	0.40861	3071	0.29608	0.40647	37.134%
Complex-5	1822	0.38176	3063	0.30815	0.38341	37.298%
Simple-10	1721	0.40643	3169	0.29912	0.40045	35.194%
Complex-10	1813	0.39810	3077	0.30278	0.39349	37.076%
Simple-20	1715	0.37852	3165	0.31121	0.38151	35.143%
Complex-20	1990	0.36823	2890	0.31601	0.37030	40.779%

Figure 3: Summary of results and outputs of test set predictions

Confusion Score	0 - "Not Confused"			1 - "Maybe Confused"			2 - "Confused"		
	Correct	Wrong	% Correct	Correct	Wrong	% Correct	Correct	Wrong	% Correct
SNN	674	1337	33.416%	80	1366	5.533%	890	536	62.412%
Simple-5	1345	668	66.816%	108	1338	7.469%	361	1065	25.316%
Complex-5	1397	616	69.399%	132	1314	9.129%	293	1133	20.547%
Simple-10	1106	911	54.834%	120	1327	8.293%	495	931	34.712%
Complex-10	1531	486	75.905%	12	1434	0.898%	269	1157	18.864%
Simple-20	967	1041	48.157%	51	1396	2.524%	697	728	48.912%
Complex-20	1890	118	94.123%	0	1447	0%	100	1325	7.017%

Figure 4: Breakdown of test set predictions by confusion score

Confusion	simple, 20	complex, 20	simple, 10	complex, 10	simple, 5	complex, 5	SNN
0	0.311	0.355	0.309	0.356	0.24	0.313	0.286
1	0.333	0.296	0.315	0.304	0.299	0.281	0.328
1	0.293	0.298	0.271	0.304	0.26	0.273	0.314
1	0.272	0.298	0.236	0.3	0.204	0.252	0.23
1	0.283	0.297	0.261	0.293	0.262	0.28	0.318
1	0.285	0.295	0.297	0.293	0.3	0.265	0.333
1	0.292	0.292	0.304	0.293	0.318	0.255	0.33
1	0.299	0.29	0.31	0.294	0.332	0.249	0.339
1	0.333	0.289	0.478	0.295	0.244	0.25	0.364
1	0.323	0.289	0.317	0.296	0.315	0.287	0.338
1	0.325	0.29	0.318	0.297	0.33	0.275	0.34
0	0.351	0.358	0.381	0.365	0.379	0.393	0.278
1	0.289	0.293	0.282	0.297	0.258	0.227	0.181
1	0.248	0.295	0.244	0.297	0.214	0.203	0.291
1	0.268	0.302	0.242	0.293	0.223	0.27	0.29
1	0.246	0.298	0.22	0.293	0.207	0.238	0.291
1	0.238	0.294	0.213	0.293	0.203	0.211	0.291
1	0.234	0.291	0.21	0.294	0.201	0.193	0.291
1	0.232	0.288	0.208	0.295	0.2	0.181	0.29
1	0.231	0.285	0.207	0.296	0.224	0.27	0.291
1	0.231	0.283	0.207	0.297	0.207	0.238	0.29
1	0.264	0.28	0.2	0.298	0.232	0.214	0.19
0	0.383	0.374	0.416	0.356	0.393	0.428	0.273
1	0.233	0.276	0.218	0.298	0.206	0.182	0.291
1	0.232	0.274	0.241	0.293	0.224	0.27	0.291
1	0.264	0.273	0.225	0.293	0.239	0.242	0.19
1	0.266	0.272	0.234	0.294	0.24	0.218	0.191
1	0.233	0.271	0.224	0.294	0.209	0.197	0.291
1	0.265	0.27	0.222	0.295	0.233	0.184	0.19
1	0.267	0.27	0.235	0.295	0.26	0.28	0.19
1	0.234	0.27	0.224	0.296	0.213	0.247	0.291
1	0.232	0.27	0.216	0.296	0.206	0.218	0.291
1	0.231	0.27	0.212	0.296	0.203	0.197	0.291
0	0.385	0.381	0.422	0.359	0.394	0.433	0.273
1	0.303	0.302	0.259	0.294	0.261	0.28	0.19
1	0.255	0.298	0.23	0.293	0.213	0.247	0.291
1	0.241	0.295	0.217	0.293	0.206	0.218	0.291
1	0.235	0.292	0.212	0.294	0.203	0.197	0.291
1	0.233	0.289	0.209	0.295	0.202	0.183	0.291
1	0.231	0.287	0.208	0.296	0.224	0.27	0.291
1	0.231	0.285	0.207	0.297	0.207	0.238	0.291
1	0.23	0.282	0.207	0.298	0.203	0.211	0.291
1	0.23	0.28	0.206	0.298	0.201	0.193	0.291
1	0.23	0.277	0.206	0.299	0.2	0.181	0.291
0	0.386	0.377	0.381	0.364	0.378	0.373	0.272
1	0.231	0.272	0.23	0.293	0.213	0.247	0.291
2	0.351	0.348	0.367	0.342	0.371	0.364	0.536
2	0.352	0.348	0.381	0.342	0.355	0.371	0.407
2	0.351	0.348	0.367	0.342	0.372	0.375	0.545
2	0.352	0.348	0.382	0.343	0.36	0.353	0.399
2	0.352	0.348	0.382	0.343	0.355	0.366	0.395
2	0.351	0.348	0.366	0.344	0.376	0.373	0.531
2	0.351	0.348	0.36	0.344	0.368	0.375	0.528
2	0.351	0.348	0.353	0.345	0.367	0.373	0.474
2	0.345	0.347	0.366	0.344	0.423	0.349	0.433
0	0.354	0.355	0.387	0.36	0.384	0.371	0.274

Figure 5: Comparison of correct answer versus the probability outputs of all models, for a subset of the test dataset

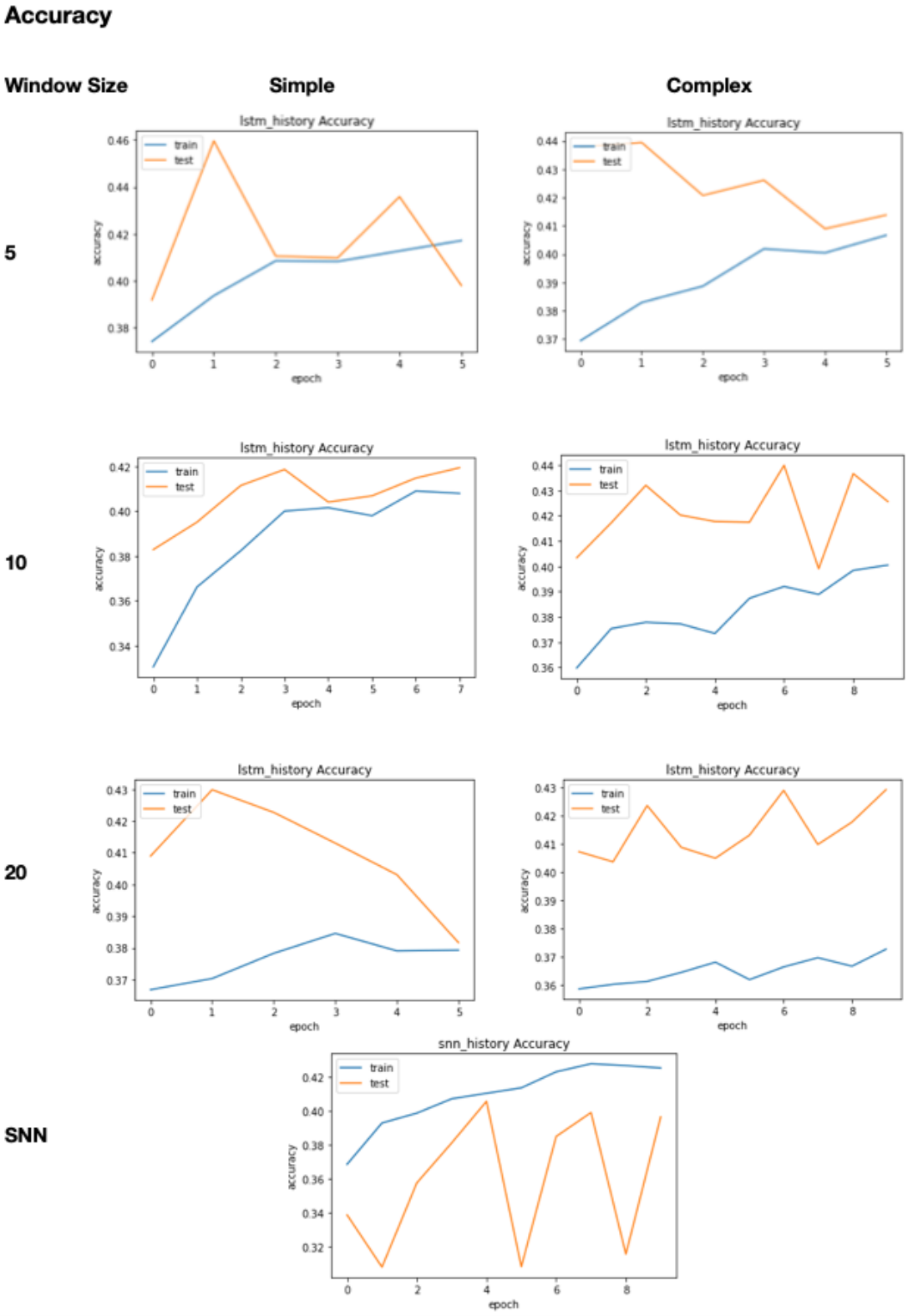

Figure 6: Accuracy graphs for all models, per epoch

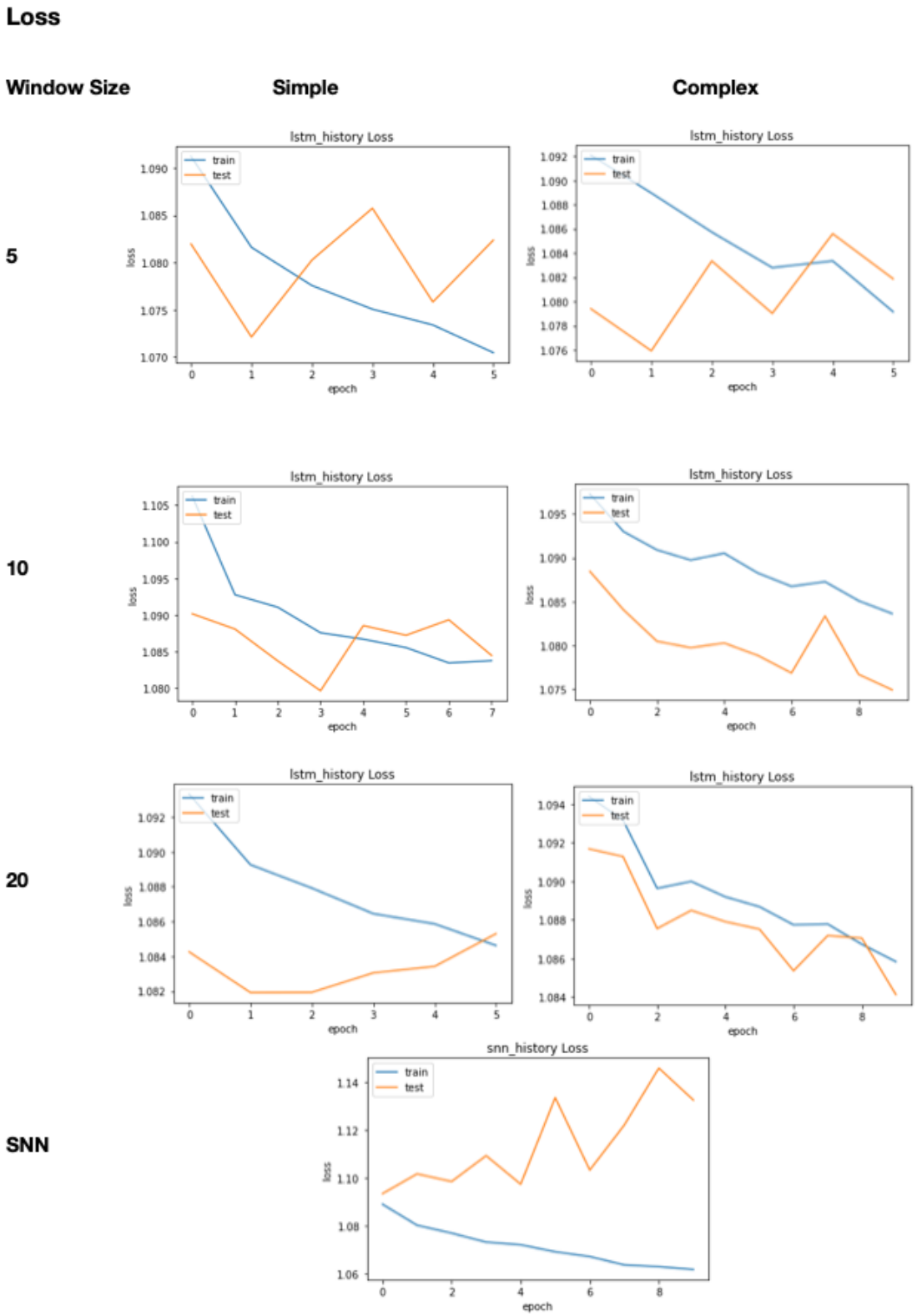

Figure 7: Loss graphs for all models, per epoch

Window Size: 5

Simple

	loss	accuracy	val_loss	val_accuracy
0	1.091238	0.374072	1.081972	0.391816
1	1.081609	0.393598	1.072126	0.459591
2	1.077571	0.408451	1.080262	0.410486
3	1.075064	0.408195	1.085747	0.409719
4	1.073394	0.412676	1.075830	0.435806
5	1.070468	0.417093	1.082371	0.397954

Complex

	loss	accuracy	val_loss	val_accuracy
0	1.092077	0.369462	1.079393	0.437852
1	1.088959	0.382907	1.075925	0.439386
2	1.085729	0.388668	1.083344	0.420716
3	1.082792	0.401857	1.079017	0.426087
4	1.083344	0.400448	1.085593	0.408951
5	1.079145	0.406658	1.081864	0.413811

10

	loss	accuracy	val_loss	val_accuracy
0	1.106244	0.330493	1.090137	0.382864
1	1.092748	0.366176	1.088048	0.395141
2	1.091040	0.382383	1.083689	0.411509
3	1.087541	0.400064	1.079601	0.418670
4	1.086666	0.401602	1.088523	0.404092
5	1.085509	0.398014	1.087199	0.406905
6	1.083416	0.409033	1.089315	0.414834
7	1.083720	0.408008	1.084431	0.419437

	loss	accuracy	val_loss	val_accuracy
0	1.097286	0.359769	1.088457	0.403325
1	1.092997	0.375336	1.084073	0.417136
2	1.090896	0.377835	1.080468	0.431969
3	1.089750	0.377194	1.079740	0.420205
4	1.090519	0.373414	1.080271	0.417647
5	1.088263	0.387316	1.078855	0.417391
6	1.086743	0.391992	1.076855	0.439898
7	1.087271	0.388853	1.083356	0.398977
8	1.085096	0.398334	1.076692	0.436573
9	1.083634	0.400448	1.074926	0.425575

20

	loss	accuracy	val_loss	val_accuracy
0	1.093310	0.366795	1.084253	0.408929
1	1.089255	0.370320	1.081921	0.429847
2	1.087913	0.378269	1.081932	0.422704
3	1.086447	0.384551	1.083046	0.413010
4	1.085858	0.379038	1.083419	0.403061
5	1.084620	0.379295	1.085297	0.381633

	loss	accuracy	val_loss	val_accuracy
0	1.094386	0.358654	1.091672	0.407143
1	1.093160	0.360256	1.091274	0.403571
2	1.089627	0.361282	1.087542	0.423469
3	1.089987	0.364487	1.088484	0.408673
4	1.089185	0.368077	1.087897	0.404847
5	1.088675	0.361923	1.087511	0.413010
6	1.087734	0.366410	1.085354	0.428827
7	1.087775	0.369679	1.087177	0.409694
8	1.086729	0.366667	1.087046	0.417602
9	1.085828	0.372692	1.084120	0.429082

SNN

	loss	accuracy	val_loss	val_accuracy
0	1.090047	0.392704	1.103531	0.388533
1	1.082303	0.405056	1.085644	0.388277
2	1.079980	0.409088	1.086737	0.401075
3	1.078545	0.413696	1.076109	0.400563
4	1.075982	0.408960	1.075545	0.406962
5	1.074249	0.412608	1.075383	0.413617
6	1.072723	0.415232	1.069430	0.405170
7	1.069554	0.417408	1.070630	0.414128
8	1.068821	0.417088	1.082105	0.411825
9	1.066891	0.420416	1.070016	0.414384

Figure 8: Output values for all models, per epoch