A Novel CNN-LSTM Model for Classification of Horizontally Interchangeable Images

Group 24

Li Xinwei Luo Chengchang Wei Jingjue

***Abstract*—This report contains motivations and basic concept of the CNN-LSTM model on image classification problem, empirical experiment on the Fashion-MNIST dataset as a representative of horizontally interchangeable images based on CNN, general CNN-LSTM model, and novel aggregated CNN-LSTM model, and results including both quantitative metric confusion matrix & accuracy and visual metric AUC &t-SNE plots, with aiming of demonstrating the overperformance of our novel CNN-LSTM model for horizontally interchangeable images.**

I. Introduction

A. Motivation

For the Image Classification problem, on the one hand, Convolutional Neural Network (CNN) has a leading performance for a very long time, which also inspires the latest state-of-art more complicated deep neural networks. The key of the CNN architecture is called the convolutional layers or filter layers, who contribute to the CNNs advantage of capturing local blocks' features from the image.

On the other hand, Long Short Term Memory (LSTM), a pervading deep neural network for sequential data analysis and prediction, sometimes is also used in computer vision. Because an image is comprised of pixels so that it can be regarded, from top to bottom and from left to right, as a sequence of pixels with a value of RGB or grey. The timesteps while converting to the sequential data are determined case by case. Whether a pixel is an individual timestep or a row of the pixels is an individual timestep is all possible. Thanks to the Long Short Term Memory cell, LSTM has the advantage that considers the covariation among different locations of the image.

In order to combine the advantages of both CNN and LSTM, CNN+LSTM concept are widely researched recently on image classification problem. In general, the images are only divided into two groups on whether images are independent or a temporal sequence(such as the different timesteps image of an arterial). For independent image dataset, CNN-LSTM structure are mostly used by researchers. However, no further division are discussed.

B. Research Gap and Project Objectives

Intuitively, images datasets can also be divided into two subgroups: horizontally interchangeable images and horizontally non-interchangeable images. A canonical definition of horizontally interchangeable images is that the images won't satisfy any of the following two cases:

- Horizontal flip of the image probably changes the original label of image like handwriting numbers.
- Horizontal flip of the image increases unnecessary noise for the image classification like X-ray of chest.

This project only focuses on the the classification problem on horizontally interchangeable images with which CNN-LSTM structure still is a potential tool to deal. However, there are two current research gaps:

- The performance of CNN+LSTM structure on horizontally interchangeable image is rarely discussed. Majority research dataset are either temporal sequential images [14] [15] [17] or independent images that is horizontally non-interchangeable [16] [18] [19].
- No specific development of CNN+LSTM structure based on horizontally interchangeable image properties is made.

To address the gap, this research attempts to achieve the following two objectives.

- Verify the effectiveness of CNN+LSTM structure compared with pure CNN for horizontally interchangeable image classification.
- Develop a novel CNN+LTSM structure for horizontally intrchangeable image classification so that its performance can be better than that of the general CNN+LSTM model.

II. Background and Related Work

A. CNN

Convolutional neural network (CNN) is a kind of multi-layer neural network. Y. LeCun in reference [2] proposed the architecture of LeNet-5 (one kind of CNN), which is shown in Fig.1.

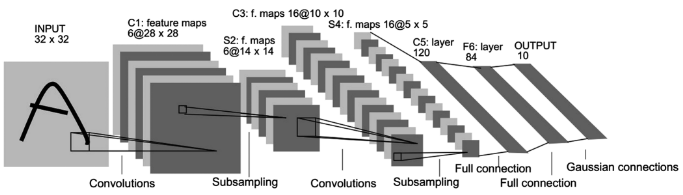


Fig. 1: Construction of LeNet-5

A CNN usually consists of three layer types: Convolutional layer, Pooling (Subsampling) layer, and Fully-Connected layer:

- *Convolutional Layer.* This is a core part of CNN, aiming to abstract features from the input images [3]. The convolutional layer contains multiple feature maps, which are extracted from the input images by convolution filters (kernels). Each filter is a weight matrix with size of 3x3 or 5x5 (usually odd dimensions), and since there are several layers in a CNN, by moving the filter with a given step parameter, we can obtain local characteristics of different positions from the former layer. Equation (1) and (2) here show the formulas when choosing filter number as 1 and step parameter as 1, where $a^{[l-1]}$ is the output of $(l-1)$ layer ($a^{[0]}$ is the original input images), $W^{[l]}$ is the filter of l layer, $b^{[l]}$ is the bias of l layer. After the filter step, the ouputs are passed into a nonlinear activation function. Common activation functions include sigmoid, tanh, and ReLu. As shown in equation (3), g is an activation function, obtaining $a^{[l]}$ as the input of next layer.

$$z^{[1]} = W^{[1]}a^{[0]} + b^{[1]} \tag{1}$$
$$z^{[l]} = W^{[l]}a^{[l-1]} + b^{[l]} \tag{2}$$
$$a^{[l]} = g(z^{[l]}) \tag{3}$$

- *Pooling Layer.* The pooling layer is usually placed between two convolutional layers [4]. whose main purpose is to gradually reduce the spatial size of the data, so that we can reduce the number of parameters in the network, computational cost and effectively control overfitting. The most common way is using a 2x2 filter with a step size of 2 to reduce the dimensions of feature maps. The typical pooling operations are average pooling (take average of the filter space) and max pooling (take the max value of filter space).
- *Fully-connected Layer.* After the process of several convolutional and pooling layers, we have obtained the feature maps of the input images. Then all the neurons in the former layer are connected with each single neuron in fully-connected layer [5]. Finally, the last fully-connected layer is followed by an output layer, and the outputs are used to do classfication tasks, where softmax regression is commonly used.

Recalling the LeNet-5 in figure 2, it consists of 2 convolutional layers and 2 pooling (subsampling) layers, and 2 fully-connected layers, which are the basic components of CNN.

B. LSTM

Recurrent Neural Network (RNN) is initially designed for capturing temporal features of sequential data in David Rumelhart's work in 1986 [6]. A classic RNN is comprised of an input layer, hidden layers and an output layer. The past time-steps inputs more or less influence the current time steps output through the chain-like architecture. However, Hochreiter and and Bengio have discussed that RNNs can't handle "long-term dependencies", widely known as "the problem of vanishing gradients" later, revealing in RNN model the former time-step t's input has little weight to predict the latter time-step t''s output, if $t' - t$ is large [7], [8]. Long-Short Term Memory (LSTM) model was born for addressing this issue.

LSTM model is a derivative of RNN, which shares the same architecture except replacing hidden layer chunks with memory cells, showed in Fig. 2. This special memory cell helps LSTM to keep long-term memory and decide when is suitable to forget it. The memory cell for timestep t consists of three gates: forget gate $\Gamma_f^{<t>}$, update gate $\Gamma_u^{<t>}$ sometimes also called input gate, and output gate $\Gamma_o^{<t>}$. In addition, $x^{<t>}, a^{<t>}, y^{<t>}, c^{<t>}$, and $\tilde{c}^{<t>}$ are the input, activation, output, cell state, and candidate cell state, respectively.

For each timestep t, the memory cell contains:

$$\Gamma_f^{<t>} = sigmoid(W_f[a^{<t-1>}, x^{<t>}] + b_f) \tag{4}$$
$$\Gamma_u^{<t>} = sigmoid(W_u[a^{<t-1>}, x^{<t>}] + b_u) \tag{5}$$
$$\tilde{c}^{<t>} = tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c) \tag{6}$$
$$c^{<t>} = \Gamma_f^{<t>} \circ c^{<t-1>} + \Gamma_u^{<t>} \circ \tilde{c}^{<t>} \tag{7}$$
$$\Gamma_o^{<t>} = sigmoid(W_o[a^{<t-1>}, x^{<t>}] + b_o) \tag{8}$$
$$a^{<t>} = \Gamma_o^{<t>} \circ tanh(c^{<t>}) \tag{9}$$

, where W, b are parameters, and $\circ$ is element-wise product.By iterating the above procedure at each time-step, we finally get the full LSTM architecture in Fig. 3 [9] [10].

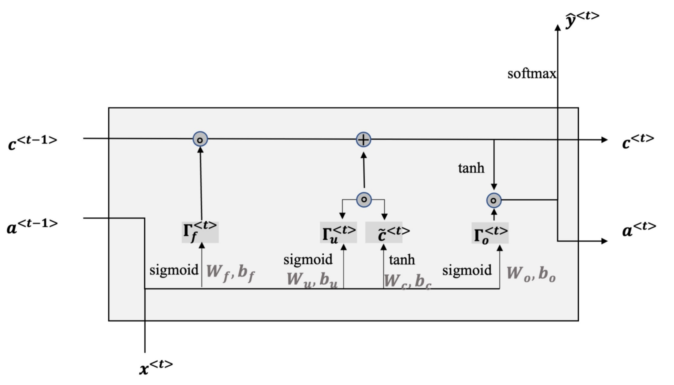


Fig. 2: Memory cell components

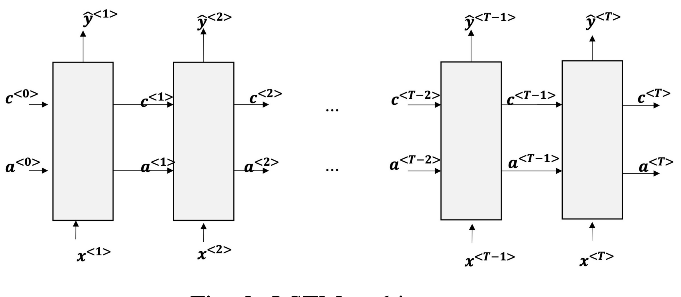


Fig. 3: LSTM architecture

There are some adaptations of LSTM to make it potent for a specific type of sequential data. For instance, Bidirectional LSTM (BLSTM) allows $X^{<t>}$ to backward affect $y^{<t-m>}$ for $m > 0$ [10]. Furthermore, adding a Conditional

Random Field (CRF) layer after the BLSTM memory cell layer (BLSTM-CRF) can receive a more robust and accurate sequence tagging result by exploiting the dependence between output sequences [11]. Meanwhile, Sutskever Neural Machine Translation (NMT) Model was created to use one layer of LSTM as an encoder to translate input into a fixed-length internal representation and let another LSTM layer decode the vectors into output values in order to deal with the unbalanced input and output length in NLP tasks. Later, Encoder-Decoder LSTM augmented with attention mechanism further removed the limitation of fixed-length of internal representation vector [12]. All of these developments and revolutions keep LSTM holding first place for traditional sequential data field like language translation, audio recognition, etc.

LSTM also has strong potential for the image classification task. Lately, Sequencer, a deep LSTM model with self-attention, can achieve state-of-the-art performance among diverse benchmark datasets compared to CNNs and even Vision Transformers [13].

C. CNN+LSTM

CNN and LSTM have unique advantages and excellent results when they are used alone. Intuitively, it's reasonable to combine them to get a hybrid version for better performance since CNN is good for feature extraction and LSTM is good for sequence prediction. The hybrid model, called CNN-LSTM, has been used in various fields, mainly dealing with classification and prediction problems.

Aditi et al. used a hybrid LSTM-CNN to deal with image classification problem [14]. In their network, after a batch normalization layer and a reshape layer, the normalized and reshaped data were passed through the LSTM cell where *Tanh* was used as activation function. Then the convolutional layer received the outputs of LSTM, extracting local important features, where *ReLU* was used as activation function. Finally, after a fully-connected layer, the final outputs were obtained. They applied their model to *MNIST* dataset and *Breast Cancer IDC* dataset, both obtaining better performances than using LSTM or CNN alone. Islam et al. applied a CNN-LSTM hybrid model to medical field. They combined CNN and LSTM to detect COVID-19 from X-ray images, where CNN extracted features from the input data, and LSTM did the classification based on these features [15]. In addition to single-label classification, this hybrid model also has good performance in multi-label classification. Hua et al. designed a class-wise attention-based convolutional and bidirectional LSTM network (CA-Conv-BiLSTM) for multi-label aerial image classification, consisting three parts: a feature extraction module, a class attention learning layer, and a Bidirectional LSTM-based recurrent sub-network, where CNN was used for feature extraction in the first part and a bidirectional LSTM-based sub-network was used to model the underlying class dependency in the third part [16]. They used UCM multi-label dataset, and not like classic classification, in their research each aerial image was categorized into more than one labels with different proportions.

CNN-LSTM hybrid model is also used widely in sequence data. Li et al. proposed a parallel LSTM-CNN network architecture to predict real-time car crash risk, where two LSTM layers and two CNN layers received the input data simultaneously and learned it separately, and then LSTM and CNN layers were combined together by a concatenate layer [17]. In their research, they considered various features, including traffic flow characteristics, signal timing, and weather conditions, which were related in chronological order. They used these features to predict the crash probabilities in the next 5-10 minutes. In medical field, this hybrid model has good performances on DNA and RNA classification. Tasdelen et al. suggested a hybrid CNN-LSTM model for pre-miRNA classification [18]. Here they used three convolution layers to extract features from the input data (spatial and sequential structure of pre-miRNA), then after a flatten layer, the outputs were passed to a LSTM layer with 100 units following a dropout layer on the fully-connected layer. Ozkok et al. proposed a hybrid CNN-LSTM model for high resolution melting(HRM) curves classification, where CNN was used for feature extraction and LSTM was used for classification of the extracted features [19]. They applied their model to HRM dataset, which was generated from DNA sequence, and obtained good results.

Among the applications of CNN-LSTM, some used a sequential CNN-LSTM model which was the most common structure, where CNN extracted features from the input data and LSTM did classificationn or prediction based on the outputs of CNN [15], [16], [18], [19]. Some used a sequential LSTM-CNN model, where LSTM layer was put before the convolutional layer to improve the performance [14]. And some used paralell CNN-LSTM model where CNN and LSTM layers as separate components, dealing with input data simutaneously, and combined their outputs together in later steps [17]. Fig.4 show these three CNN-LSTM models, and all the combinations of CNN and LSTM outperformed other approaches only using one of them.

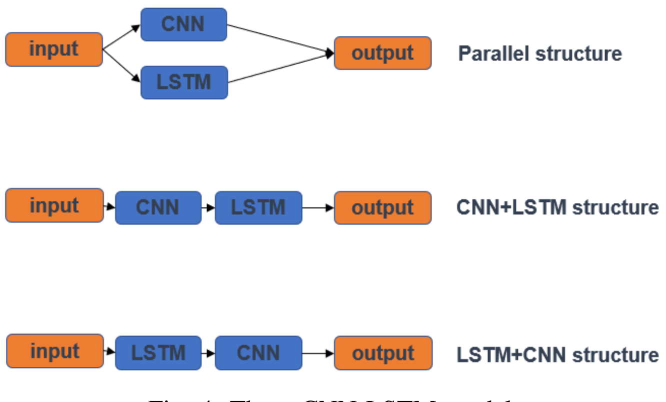


Fig. 4: Three CNN-LSTM models

III. Experiment Design

A. Assumptions

The research uses Fashion-MNIST as a representative for horizontally interchangeable image classification data sets and assumes that the model and successful performance will transfer to other horizontally interchangeable image data sets as well.

B. Dataset Chosen and Exploration

1) Dataset Chosen: Recalling the datasets used in the references, [17]–[19] used sequence data which had strict requirements on the order, which didn't apply to our project. Although [14]–[16] used images as input, their result may be affected by horizontal flip because the information may change.

Fashion-MNIST dataset, based on the assortment on Zalando's website, contains 28x28 grayscale images of 70000 unique fashion products, coming from different gender groups: men, women, kids and neutral. All 70000 images are chosen from front look thumbnail images, which are classified into 10 classes with 7000 images per class: T-Shirt/Top, Trouser, Pullover, Dress, Coat, Sandals, Shirt, Sneaker, Bag, and Ankle boots. Fig.5 from reference [20] shows the classification and the pictures of each class.

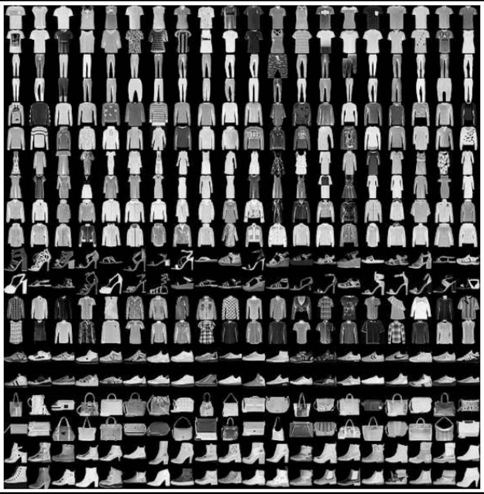
Label	Description	Examples
0	T-Shirt/Top	
1	Trouser	
2	Pullover	
3	Dress	
4	Coat	
5	Sandals	
6	Shirt	
7	Sneaker	
8	Bag	
9	Ankle boots	

Fig. 5: Classes of Fashion-MNIST dataset

We perform our method on Fashion-MNIST dataset, and there are four main reasons for us to choose it:

- *Not too simple*. From this t-SNE visualization on raw data in Fig.6, it's obvious that most labels are clustered well but still overlap each other sometimes, which indicated that an advancing image classifier for this task is necessary.
- *Horizontally interchangeable images*. It would not be affected by horizontal flip. Therefore, we can extend our dataset in training and test sets by horizontal flip to increase the final accuracy.

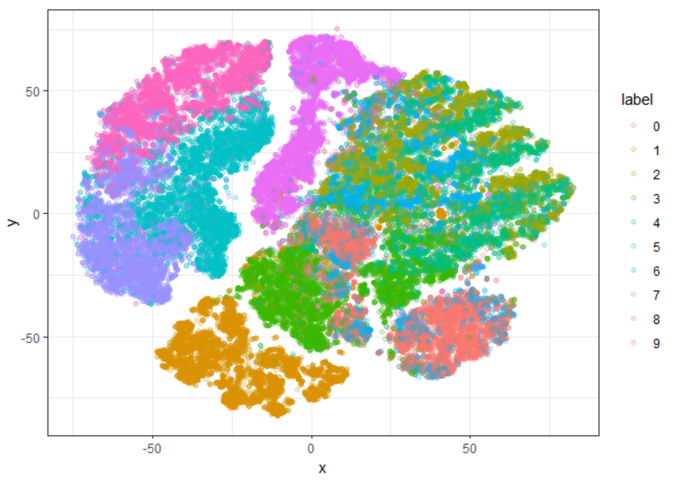


Fig. 6: t-SNE of raw data

- *Balanced*. There are 60000 examples in training set, where each label has 6000, and 10000 examples in test set, where each label has 1000.
- *Open source*. Fashionmnist is a well-known and excellent dataset for testing image classification methods. There are dozens of algorithms tested on it and post their accuracy publicly.This makes it easy to compare our results with other methods in previous papers.
  The highest accuracy currently goes to the Dual-path network with wide resnet 28-10, whose accuracy is 95.7% (without validation) [22]. Given the performance of a method is strongly affected by its computing resource and running time, we only list those methods using similar computing resources as ours with a leading accuracy in Table I.

2) Explorative Research: We do some explorations on Fashion-MNIST. Here we show two box plots (Fig.7, Fig.8) of mean and variance of total 70000 examples in each label, respectively. We can see that the means and variances are different among the labels, especially between labels of shoes (E.g., Sandal & Sneaker) and labels of clothes (E.g., T-shirt/top & Pullover), since their pixel grey values are different, which depends on the volume of the object. The means of T-shirt in the label 0 and shirt in the label 6 are similar since they usually only differ in the length of sleeves, so it's more difficult to distinguish them than others, which is one of our challenges.

IV. Data Modelling, Evaluation, and Result

A. Model Structure

In this project, we totally construct six models. We first construct two CNN models with the same layer structure, but different dimensions as the baseline (Model III & IV). The main structure is that we develop two convolutional layers with two max-pooling layers, followed by two fully connected layers. Finally, the output is the result of the softmax function. Then, based on these two baselines, we add one LSTM layer into these two models with two different dimensions (Model

TABLE I: Benchmark method perfromance on Fashionmnist

Model	Structure and Methods	Accuracy
Human Performance	1000 random sample per person; 3 label choice each image; No fashion expert;	83.50% [22]
CNN+LSTM	Cov1: $(3, 3, 20)$; [a] Maxpooling: $(2, 2)$; Cov2: $(5, 5, 20)$; LSTM + FC + softmax. Optimizer: Maximum Likelihood.	91.36% [1]

CNN	Cov1: $(5, 5, 32)$,
	padding=same;
	Maxpooling: $(2, 2)$,
	stride = 2;
	Cov2:$(5, 5, 64)$;
	Maxpooling: $(2, 2)$
	stride = 2;
	FC+ softmax.
	Optimizer: GradientDescentOptimizer;
	Batchnorm and dropout are used.
		91.60%
		[22]

CNN Model with SVM Classifer[b]	Cov1: $(5, 5, 32)$; Maxpooling: $(2, 2)$; Cov2:$(5, 5, 64)$; Maxpooling: $(2, 2)$; FC + SVM. Batchnorm and dropout are used.	90.72% [23]

[a](m,n,k) denoted the $m \times n$ kernel whose kernel number is k. The padding and stride default are 0 and 1, which will be specifically annotated if not the default value.

[b] SVM is short for Support Vector Machine

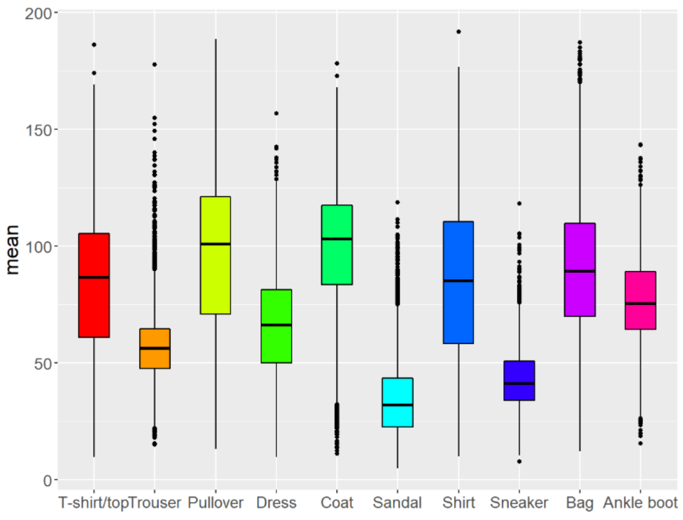


Fig. 7: Box plot of means

I & II). After that, we modify input and output methods on Model I and II as Model V and VI. Table II are the results and structures of these six models.

The best example model architecture is the structure of the model VI. The details are shown in details on the Fig.9. The working flow is that we first input our image data into one convolutional layer plus one max pooling layer, twice. Then, we use the view function to flatten our results, and put them into a fully connected layer, following with the LSTM layer. After that, there is another fully connected layer to receive the output of the LSTM layer, and we finally modify the input and output using random flip and aggregated output, respectively.

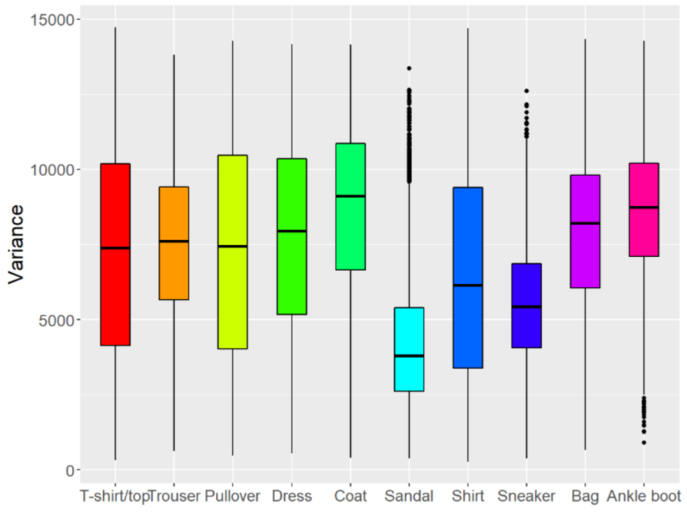


Fig. 8: Box plot of variances

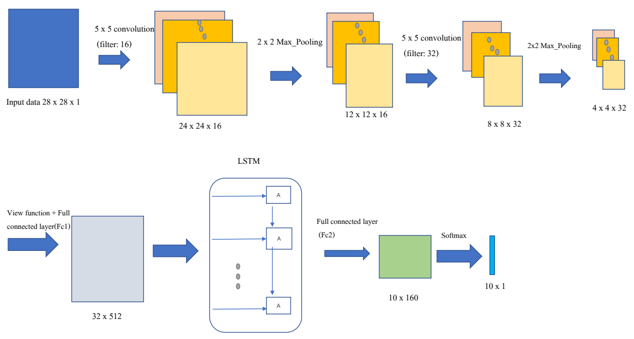


Fig. 9: Model II Structure

B. Evaluation Metrics

All models are compared by four metrics Accuracy, F1 scores, AUC and t-SNE visualization. In addition, we modify our model with two improvements: random flip and aggregated output.

1) Accuracy: For each class $i = 1, 2, \cdots, m$:

$$TP_i = T_i \tag{10}$$

$$TN_i = \Sigma_{j \neq i} T_i + \Sigma_{j \neq k \neq i} F_{jk} \tag{11}$$

$$FP_i = \Sigma_{j \neq i} F_{ij} \tag{12}$$

$$FN_i = \Sigma_{j \neq i} F_{ji} \tag{13}$$

$$Accuracy_i = \frac{TP_i}{TP_i + FN_i} \tag{14}$$

$$Recall_i = \frac{TP_i}{TP_i + FP_i} \tag{15}$$

$Accuracy_i$ and $Recall_i$ are class based mean accuracy and recall [21].

TABLE II: Models Structure and Performance

Model	Structure[a]	Accuracy
Model I	Conv1: (1,16,5); Maxpooling: (2,2); Conv2: (16,32,5) Maxpooling: (2,2); FC1 : (512,32) LSTM: (32, 160, 1) FC2: (160,10) Softmax;	89.76%
Model II	Conv1: (1,8,5); Maxpooling: (2,2); Conv2: (8,32,5) Maxpooling: (2,2); FC1 : (512,64) LSTM: (64, 320, 1) FC2: (320,10) Softmax;	90.38%
Model III	Conv1: (1,8,5); Maxpooling: (2,2); Conv2: (8,32,5) Maxpooling: (2,2); FC1 : (512,64) FC2: (64,10) Softmax;	86.55%
Model IV	Conv1: (1,16,5); Maxpooling: (2,2); Conv2: (16,32,5) Maxpooling: (2,2); FC1 : (512,32) FC2: (32,10) Softmax;	86.28%
Model V	Random Flip Model I Aggregated Output	90.82%
Model VI	Random Flip Model II Aggregated Output	91.43%

[a] Model layers' structure follows Pytorch API standard Conv(in_channels,out_channels, kernel_size)
The padding and stride default are 0 and 1

TABLE III: Confusion Matrix of multi-class m

	Predicted condition				
Actual condition		**P1**	**P2**	$\cdots$	**Pm**
	1	T_1	F_{12}	$\cdots$	F_{1m}
	2	F_{21}	T_2	$\cdots$	F_{2m}
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	m	F_{m1}	F_{m2}	$\cdots$	T_m

In total,

$$TP = \Sigma_i TP_i \tag{16}$$

$$FP = \Sigma_i FP_i \tag{17}$$

$$FN = \Sigma_i FN_i \tag{18}$$

$$Accuracy = Recall = \frac{TP}{TP + FN} \tag{19}$$

The Accuracy of the total multi-class model inherits a similar idea from the two-class accuracy, only if the Precision, Recall, and Accuracy are the same in the multi-class situation.This metric is used in both external related model comparison with our models and internal comparison among our models, While the last two metrics, AUC and t-SNEs, are only used for internal comparison [19].

2) F1 score:

$$F1 = \frac{2Accuracy \times Recall}{Accuracy + Recall} = Accuracy = Recall \tag{20}$$

Micro F1 takes the same value with total Accuracy in multi-class classification.

3) AUC: Receiver Operating Characteristics (ROC) Curve is a curve of True Positive Rate (TP Rate) against False Positive Rate (FP Rate), which means the probability of the model to discriminate between right cases and wrong cases. The bigger Area Under ROC (AUC) is, the better classifier the model is. Normally, plotting AUCs together will easily get the conclusion of which model is the best. Furthermore, we can plot AUC stratified by class to compare their performances.

4) t-SNEs: t-Distributed Stochastic Neighbor Embedding (t-SNE) is a powerful visual method illustrating model's capacity of classification. It offers an outlook of model's performance and reveals individual level information [22].

However, t-SNE is facing the accuse of uncertain and possible false clustering each time because of randomness initialization. [26] To address the uncertainty, preserving global structure, and ensure the accuracy of plot as much as possible, we specially take the following techniques.

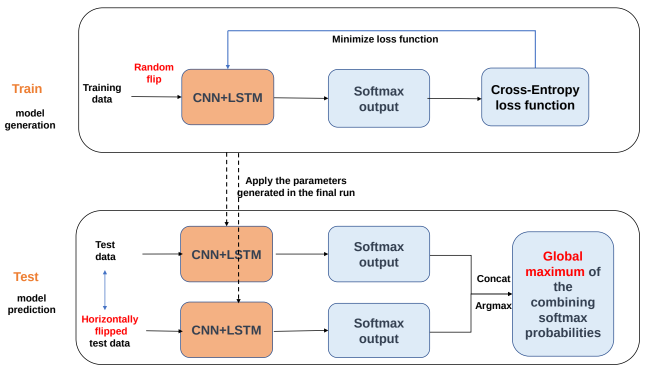
- Instead of the default random initialization at the exaggeration stage of t-SNE, We apply Principle Component Analysis (PCA) for informative initialization, which can keep a faithful global data structure and can be realized by the Python module open-tsne now. [27] [25]
- Given the strong dependence between the t-SNE performance and its parameter choosing, We set the exaggeration stage iteration number as $K_0 = \lfloor (log(n))^2 \rfloor$ to avoid "overshooting", where n is the data size. [26]
- Accuracy and interpretability metrics are applied to ensure the tSNE plots of the models are equally in compare. [28]

C. Improvement

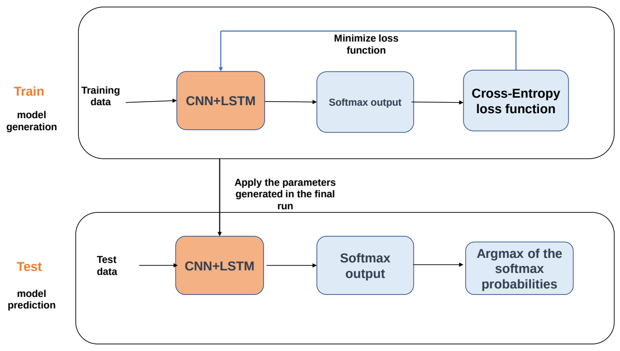
Model V & VI are CNN-LSTM aggregated models owning the following two improvement compared to Model I & II,general CNN-LSTM Models. The modelling work flow chart compirison are shown in Fig.10.

1) Horizontally Random Flip: In model evaluation phrase, each image has 50% probability to flip horizontally before input in each batch of each epoch. Therefore, we almost augement the training data set twice as much as the CNN-LSTM general model evaluation.

2) Aggregated Output: In model prediction phrase, for each input image, we combine the softmax output of the original data and the horizontally flipped data together, and take the label with global maximum probability as the predict label.



(a) Aggregated Flow Chart



(b) General Flow Chart

Fig. 10: Modelling Flow Chart Comparision

For example, as the Fig.11 shown, we first horizontally flip the original image of shoe, and then put them into our models separately to get softmax output. Last but the most important, rather than regarding the label with biggest probability in each output vector as the predict labels,label 5 and label 7 here, we concat two softmax outputs and return the label 7 as the image prediction label, since it has global maximum probability.

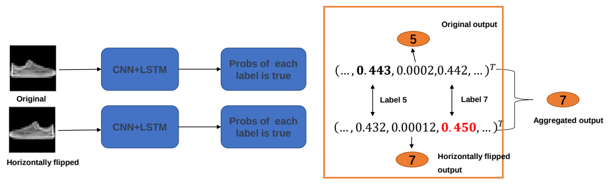

Original
Horizontally flipped
CNN+LSTM
CNN+LSTM
Probs of each label is true
Probs of each label is true
Original output
5
(..., 0.443, 0.0002,0.442, ...)^T
Label 5
Label 7
(..., 0.432, 0.00012, 0.450, ...)^T
7
Horizontally flipped output
7
Aggregated output

Fig. 11: Details of Improvement Meathod

D. Result

Based on the performance, we can see that the two CNN-LSTM models are generally better than single CNN models. Therefore, we have achieved our research objective one. In addition, we want to find out our model's prediction for each class label. Therefore, we draw the confusion matrix of our best model—Model II (Fig.12). And, we get that the worst class is the class six–shirt. Therefore, we draw the AUROC (area under the ROC curve) (Fig.13) in order to see if our model can perform well in the worst class. The result is 0.97, which means our model is very accurate.

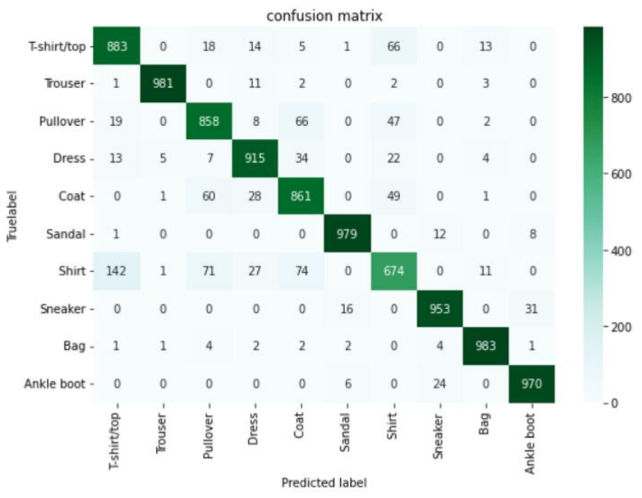


Fig. 12: Confusion Matrix of Model II

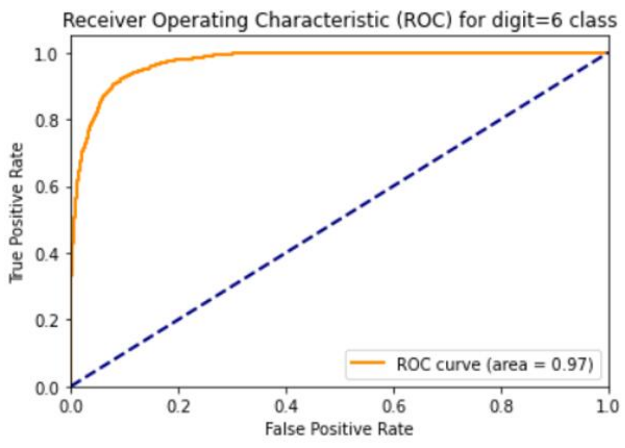


Fig. 13: AUROC of Model II

Then, we use t-SNE method to visualize our result of Modle II (Fig.14). From the picture, we can see that they are generally separated, even though there is still some overlaps.

Although our model II has already achieved 90.38% performance, there are still a lot of mis-labeled data (Fig.15). The half of samples in the picture are mis-labeled. Therefore, more techinques are needed to improve classification performance.

After we use these two improvement methods, our model VI performance achieved 91.43%, which is the best. Our

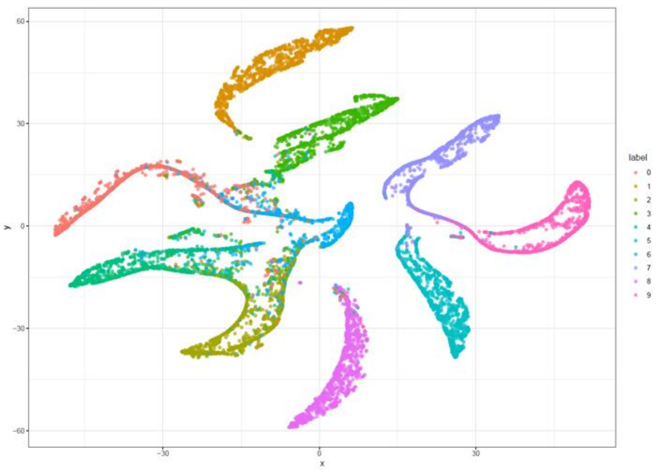


Fig. 14: t-SNE Visualization for Model II

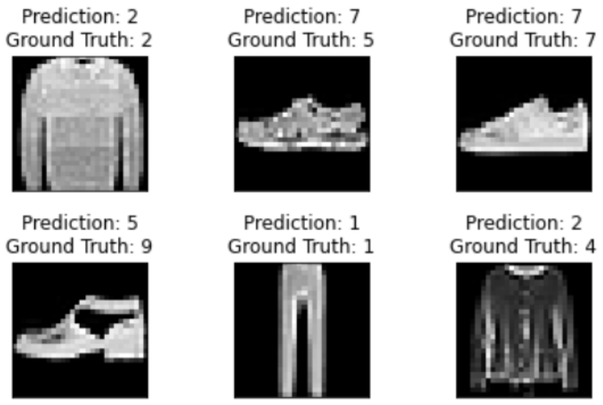


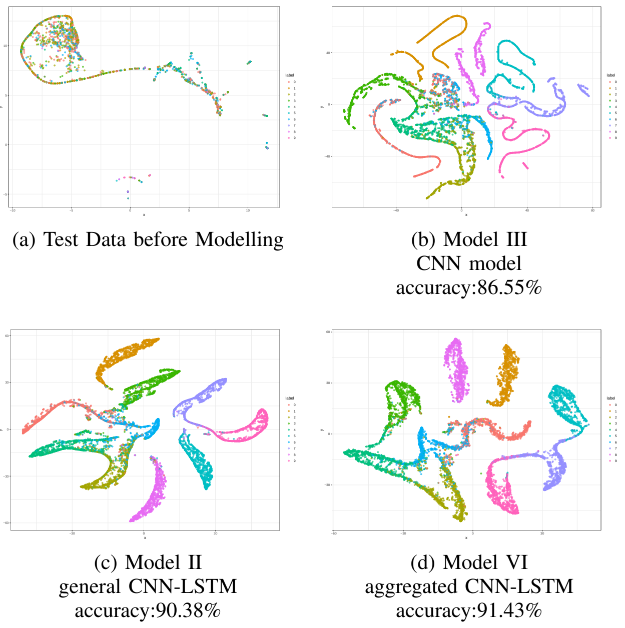
Fig. 15: Mis-labeled Data Sample

aggregated model and general model work flow chart is shown on the Fig.10. As it is shown, the differences are mainly about the data pre-processing and output modification.

At last, we use two methods to improve our model performance. Maintaining the same architecture of CNN-LSTM model, we modify the modelling work flow both in Model Generation and Model Prediction.

In conclusion, Fig.16 shows the outcomes of our entire experiment. The obvious clustering difference can be noticed between Model III and Model II, indicating the effectiveness of CNN-LSTM model for horizontally interchangeable images. Model VI t-SNE plot also has less overlapping cases compared to Model II, especially in label 1 (yellow), label 5 (green-blue), and label 8 (purple). This visual difference easliy shows our CNN-LSTM model overperforms general CNN-LSTM model.

In addition, label 6 (blue) and label 0 (red) are always overlapping in all models to a different degree. Since label 6 represents shirts and Label 0 represents T-shirts, which are quite confusing types of cloth in reality. Once the model accuracy is higher than the people performance 83.5%, the improvement is far tougher. The training dataset also has certain probability to mislabel these two types, Therefore, we focus on the entire prediction accuracy rather than certain label.


(a) Test Data before Modelling

(b) Model III
CNN model
accuracy:86.55%

(c) Model II
general CNN-LSTM
accuracy:90.38%

(d) Model VI
aggregated CNN-LSTM
accuracy:91.43%

Fig. 16: tSNE plots Comparison among Different Models.

V. Conclusion and Discussion

This research contributes to our understanding of classification problem of horizontally interchangeable images , with the aim at verifying the general CNN-LSTM model effectiveness compared to CNN model with the same architecture and improving the classification accuracy by developing a novel CNN-LSTM aggregated model.

In summary, our results and contribution mainly contains three aspects:

1) Novel Guide for classification Problem of Interchangeable Images:: Horizontally interchangeable image classification problem is always ignored. Fortunately, we offer a novel CNN-LSTM model making use of image's properties to improve the classification performance without adding more computational time by combining horizontally randomly flip in model evaluation phrase and aggregated output in model prediction.

2) Better Visualization of Results:: Introducing t-SNE plots with PCA Initialization to show the visual results without loss of the rigorousness.

3) Best Performance:: Among all CNN-LSTM models on Fashion-MNIST dataset, our model obtains the best results, whose accuracy is 91.43%. The former best performance of CNN+LSTM on Fashionmnist data set is 91.36% by Ji, Y. (2022, June) [1].

Admittedly, there are still some unsettled issues on the CNN-LSTM model for the classification of horizontally interchangeable image research. Instead of our aggregated model introduced above, there is another intuitive way to do a similar thing. That is, using Aggregated Input (input image and its horizontally interchangeable image as a pair) to evaluate the CNN-LSTM model parameters and treating the label with individual softmax biggest probability as prediction label. The reason why not use aggregated output here is that the loss function is calculated pairs by pairs, as long as the parameter is approximate to the true value, the output of an image and its horizontally flip image are almost the same. However, this modification involves loss function establishment and parameter evaluation, whose code modification is far more complicated than ours. That is the reason why we choose CNN-LSTM model with random flip and aggregated output. But Whether our model has a better performance than this modified model is unknown, therefore, further research can be done in this direction.

VI. Credit Authorship Contribution Statement

Li Xinwei: Conceptualization, Methodology, Coding - *CNN-LSTM (aggregated) & t-SNE*, Writing - *introduction & literature & modelling & conclusion*.

Luo Chengchang: Coding - *model code annotating*, Writing - *literature & EDA*, Group Correspondent with TA and the Professor.

Wei Jingjue: Conceptualization, Coding - *CNN & CNN-LSTM (general) & AUC*, Writing - *modelling*.

VII. Acknowledgment

The authors would like to appreciate Prof. Markus Kirchberg for the generous, professional, and detailed review and feedback and TA Dr. Wong Yean Ling for the timely and kind help.

VIII. Resources

All datasets, codes and figures of this research can be found in github: https://github.com/ztt1/ST5188.

References

[1] Ji, Y. (2022, June). A novel CNN+ LSTM classification model based on fashion-MNIST. In *International Conference on Neural Networks, Information, and Communication Engineering (NNICE)* (Vol. 12258, pp. 178-184). SPIE.

[2] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

[3] Guo, T., Dong, J., Li, H., and Gao, Y. (2017, March). Simple convolutional neural network on image classification. In 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA) (pp. 721-724). IEEE.

[4] Wang, W., Yang, Y., Wang, X., Wang, W., and Li, J. (2019). Development of convolutional neural network and its application in image classification: a survey. Optical Engineering, 58(4), 040901.

[5] Sharma, N., Jain, V., and Mishra, A. (2018). An analysis of convolutional neural networks for image classification. Procedia computer science, 132, 377-384.

[6] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Nature, vol. 323, no. 6088, pp. 533–536, Oct. 1986, doi: 10.1038/323533a0.

[7] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Juergen Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In S. C. Kremer and J. F. Kolen, eds., A Field Guide to Dynamical Recurrent Neural Networks. IEEE press, 2001.

[8] Y. Bengio, P. Simard and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," in IEEE Transactions on Neural Networks, vol. 5, no. 2, pp. 157-166, March 1994, doi: 10.1109/72.279181.

[9] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[10] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," Neural Networks, vol. 18, no. 5–6, pp. 602–610, Jul. 2005, doi: 10.1016/j.neunet.2005.06.042.

[11] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF Models for Sequence Tagging." arXiv, Aug. 09, 2015. Accessed: Sep. 03, 2022. [Online]. Available: http://arxiv.org/abs/1508.01991

[12] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks." arXiv, Dec. 14, 2014. Accessed: Sep. 03, 2022. [Online]. Available: http://arxiv.org/abs/1409.3215

[13] Y. Tatsunami and M. Taki, "Sequencer: Deep LSTM for Image Classification." arXiv, May 17, 2022. Accessed: Sep. 03, 2022. [Online]. Available: http://arxiv.org/abs/2205.01972

[14] Aditi, Mayank.K.N, & Poovammal.E (2019). Image Classification using a Hybrid LSTM-CNN Deep Neural Network. *International Journal of Engineering and Advanced Technology*, Vol 8(6),1342-1348.

[15] Islam, M. Z., Islam, M. M., & Asraf, A. (2020). A combined deep CNN-LSTM network for the detection of novel coronavirus (COVID-19) using X-ray images. *Informatics in medicine unlocked*, 20, 100412.

[16] Hua, Y., Mou, L., & Zhu, X. X. (2019). Recurrently exploring class-wise attention in a hybrid convolutional and bidirectional LSTM network for multi-label aerial image classification. *ISPRS journal of photogrammetry and remote sensing*, 149, 188-199.

[17] Li, P., Abdel-Aty, M., & Yuan, J. (2020). Real-time crash risk prediction on arterials based on LSTM-CNN. *Accident Analysis & Prevention*, 135, 105371.

[18] Tasdelen, A., & Sen, B. (2021). A hybrid CNN-LSTM model for pre-miRNA classification. *Scientific reports*, 11(1), 1-9.

[19] Ozkok, F. O., & Celik, M. (2022). A hybrid CNN-LSTM model for high resolution melting curve classification. *Biomedical Signal Processing and Control*, 71, 103168.

[20] Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747.

[21] Y. Hua, L. Mou, and X. X. Zhu, "Recurrently exploring class-wise attention in a hybrid convolutional and bidirectional LSTM network for multi-label aerial image classification," ISPRS Journal of Photogrammetry and Remote Sensing, vol. 149, pp. 188–199, Mar. 2019, doi: 10.1016/j.isprsjprs.2019.01.015.

[22] Maaten, L.V.D., Hinton, G., 2008. Visualizing data using t-sne. J. Mach. Learn. Res. 9 (Nov), 2579–2605.

[23] "Fashion-MNIST".[Online].Available: https://github.com/zalandoresearch/fashion-mnist.git

[24] A. F. Agarap, "An Architecture Combining Convolutional Neural Network (CNN) and Support Vector Machine (SVM) for Image Classification." arXiv, Feb. 07, 2019. Accessed: Oct. 10, 2022. [Online]. Available: http://arxiv.org/abs/1712.03541

[25] D. Kobak and G. C. Linderman, "Initialization is critical for preserving global data structure in both t-SNE and UMAP," Nat Biotechnol, vol. 39, no. 2, pp. 156–157, Feb. 2021, doi: 10.1038/s41587-020-00809-z.

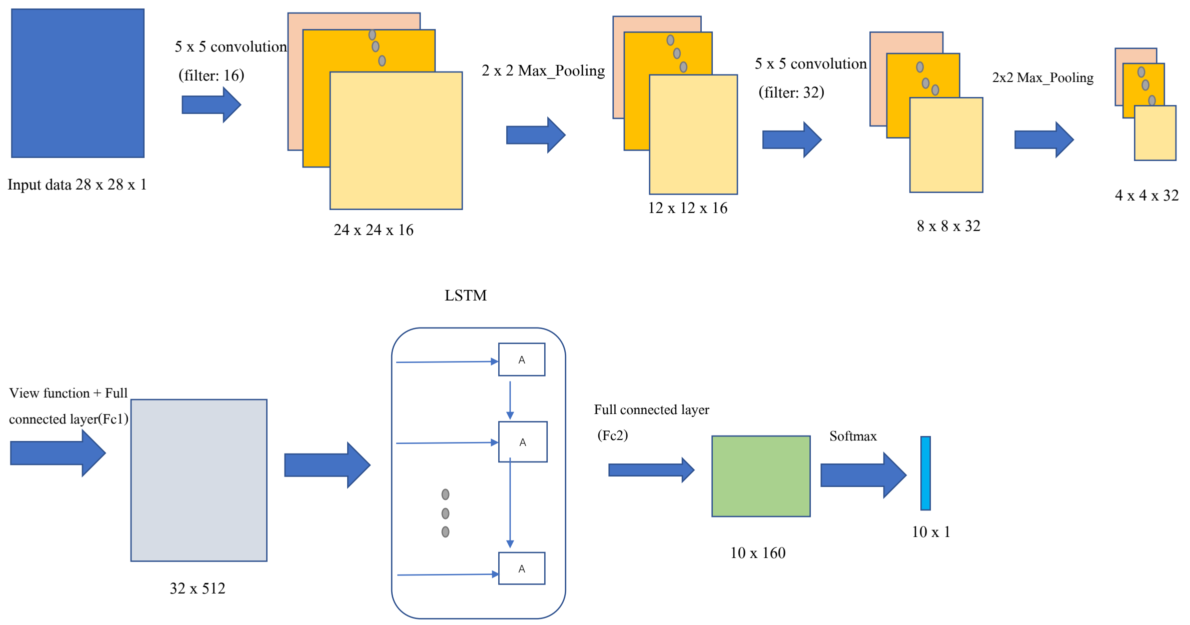
[26] T. T. Cai and R. Ma, "Theoretical Foundations of t-SNE for Visualizing High-Dimensional Clustered Data." arXiv, Mar. 28, 2022. Accessed: Oct. 10, 2022. [Online]. Available: http://arxiv.org/abs/2105.07536

[27] P. G. Poličar, M. Stražar, and B. Zupan, "openTSNE: a modular Python library for t-SNE dimensionality reduction and embedding," Bioinformatics, preprint, Aug. 2019. doi: 10.1101/731877.

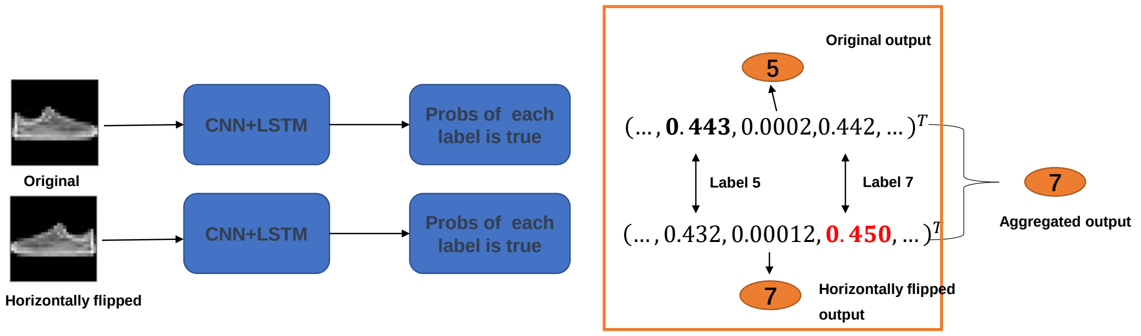
[28] R. Gove, L. Cadalzo, N. Leiby, J. M. Singer, and A. Zaitzeff, "New guidance for using t-SNE: Alternative defaults, hyperparameter selection automation, and comparative evaluation," Visual Informatics, vol. 6, no. 2, pp. 87–97, Jun. 2022, doi: 10.1016/j.visinf.2022.04.003.

IX. Appendix-High-Resolution Plots

A. CNN-LSTM Model Architecture,Fig.9

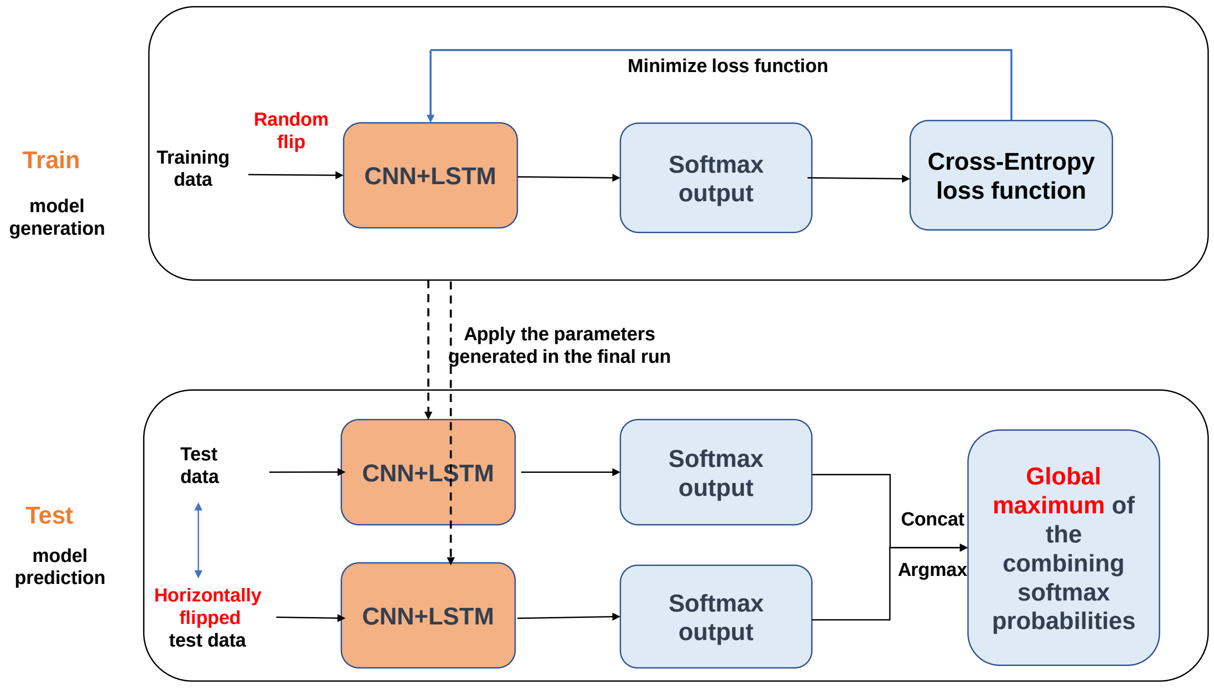


B. Improvement Example Fig.11

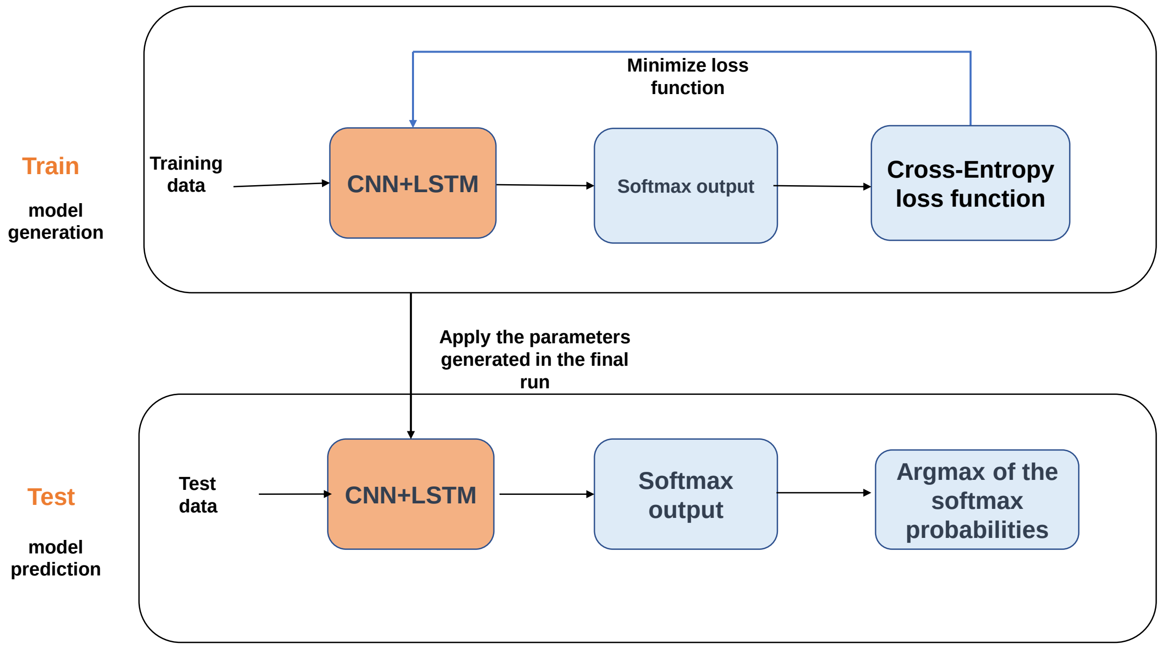


Original
Horizontally flipped
CNN+LSTM
CNN+LSTM
Probs of each label is true
Probs of each label is true
Original output
5
$(\ldots, \mathbf{0.443}, 0.0002, 0.442, \ldots)^T$
Label 5
Label 7
$(\ldots, 0.432, 0.00012, \mathbf{0.450}, \ldots)^T$
7
Horizontally flipped output
7
Aggregated output

C. Modelling Flow Chart,Fig.10

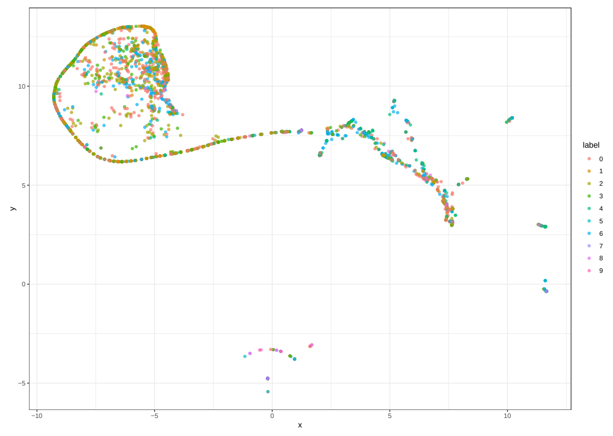


(a) Modified Flow Chart

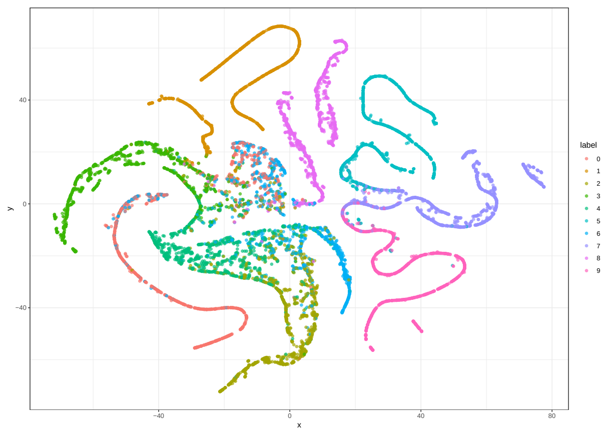


(b) General Flow Chart

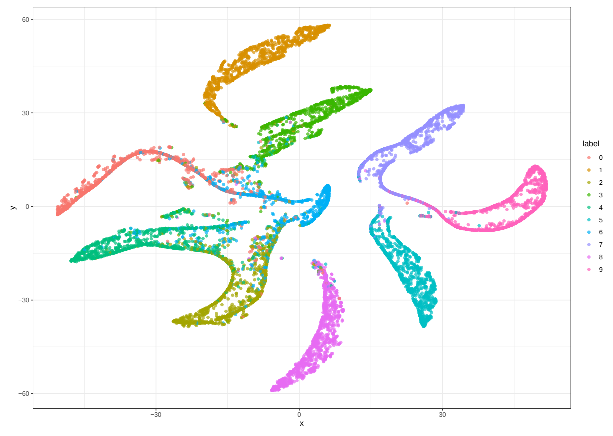
D. *tSNE plot,Fig.16*



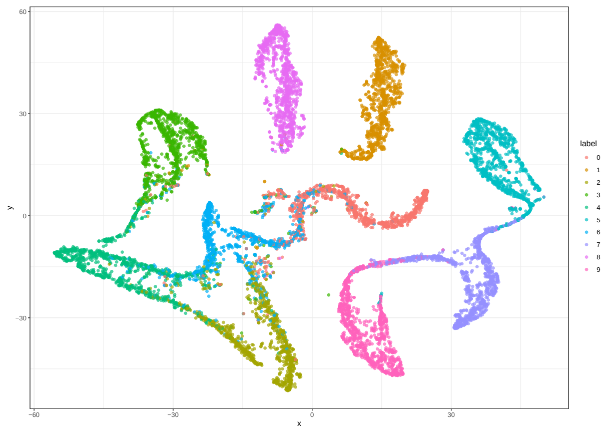
(a) Test Data before Modelling



(b) Model III
CNN model
accuracy:86.55%



(c) Model II
general CNN-LSTM Model
acccuracy:90.38%



(d) Model VI
agggregated CNN-LSTM Model
accuracy:91.43%