Metadata of the chapter that will be visualized in SpringerLink

Book Title	Human Activity Recognition Challenge	
Series Title		
Chapter Title	Cooking Activity Recognition with Convolutional LSTM Using Multi-label Loss Function and Majority Vote	
Copyright Year	2021	
Copyright HolderName	The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Singapore Pte Ltd.	
Author	Family Name	**Fujii**
	Particle	
	Given Name	**Atsuhiro**
	Prefix	
	Suffix	
	Role	
	Division	
	Organization	Graduate School of Information Science and Engineering, Ritsumeikan University
	Address	1-1-1 Nojihigashi, Kusatsu, Shiga, 525-8577, Japan
	Email	
Author	Family Name	**Kajiwara**
	Particle	
	Given Name	**Daiki**
	Prefix	
	Suffix	
	Role	
	Division	
	Organization	Graduate School of Information Science and Engineering, Ritsumeikan University
	Address	1-1-1 Nojihigashi, Kusatsu, Shiga, 525-8577, Japan
	Email	
Corresponding Author	Family Name	**Murao**
	Particle	
	Given Name	**Kazuya**
	Prefix	
	Suffix	
	Role	
	Division	
	Organization	Graduate School of Information Science and Engineering, Ritsumeikan University
	Address	1-1-1 Nojihigashi, Kusatsu, Shiga, 525-8577, Japan
	Email	murao@cs.ritsumei.ac.jp

Abstract

This paper reports the Cooking Activity Recognition Challenge by team Rit's cooking held at International Conference on Activity and Behavior Computing (ABC 2020). Our approach leverages the convolutional layer and LSTM to recognize macro activities (recipe), and micro activities (body motion). For micro activities consisting of multiple labels in a segment, loss is calculated using BCEWithLogistsLoss function in PyTorch for each body part, and then the final decision is made by majority vote by the body parts.

Cooking Activity Recognition with Convolutional LSTM Using Multi-label Loss Function and Majority Vote

Atsuhiro Fujii, Daiki Kajiwara, and Kazuya Murao

Abstract This paper reports the Cooking Activity Recognition Challenge by team Rit's cooking held at International Conference on Activity and Behavior Computing (ABC 2020). Our approach leverages the convolutional layer and LSTM to recognize macro activities (recipe), and micro activities (body motion). For micro activities consisting of multiple labels in a segment, loss is calculated using BCEWithLogistsLoss function in PyTorch for each body part, and then the final decision is made by majority vote by the body parts.

AQ1

1 Introduction

This paper reports the solution of our team "Rits's cooking" to Cooking Activity Recognition Challenge held at International Conference on Activity and Behavior Computing (ABC2020).

Activity recognition can enrich our lives by identifying the characteristics of human behavior, therefore, there have been a lot of research on human activity recognition (HAR). Pierluigi et al. [5] proposed HAR method based on accelerometer data using a wearable device. Atallah et al. [4] conducted a study on sensor positioning for HAR using wearable accelerometers. HAR using built-in sensors in smartphones is also popular. Bayat et al. [1] conducted a study on HAR using accelerometer data from smartphones. Wang et al. [2] conducted a comparative study on HAR using inertial sensors in a smartphone. Chen et al. [11] conducted a performance analysis of smartphone-sensor behavior for HAR. In addition, a device-free HAR method has been proposed by Wang et al. [10].

Recently, a lot of research on activity recognition using neural networks have been conducted, and a high degree of accuracy has been achieved. Chen et al. [13] proposed

A. Fujii · D. Kajiwara · K. Murao (✉)
Graduate School of Information Science and Engineering, Ritsumeikan University, 1-1-1 Nojihigashi, Kusatsu, Shiga 525-8577, Japan
e-mail: murao@cs.ritsumei.ac.jp

M. A. R. Ahad et al. (eds.), *Human Activity Recognition Challenge*,
Smart Innovation, Systems and Technologies 199,
https://doi.org/10.1007/978-981-15-8269-1_8

502437_1_En_8_Chapter ☑ TYPESET ☐ DISK ☐ LE ☑ CP Disp.:16/9/2020 Pages: xxx Layout: T1-Standard

a deep learning approach to HAR based on a single accelerometer. Wenchao et al. [9] conducted a study on HAR using wearable sensors by deep convolutional neural networks. Chen et al. [12] proposed an LSTM-based feature extraction approach to recognize human activities using tri-axial accelerometer data. Ordóñez et al. proposed deep convolutional and LSTM Recurrent Neural Networks for multimodal wearable activity recognition [3].

Most conventional methods consider single-label activities which means only one non-overlapping label is given to the input data. However, the cooking activity dataset we handle in this challenge includes micro- and macro activities. The micro activity consists of a set of single segments with multiple labels. In addition, the number of samples in a segment for each sensor are different.

In this paper, we constructed a network with a convolutional layer and an LSTM layer for sensors at each body part. Handcrafted features are employed as an input of the network. For micro activities, BCEWithLogistsLoss is used as the loss function to evaluate multi-label data. Four decisions obtained with the networks are then merged to one final output by majority vote.

2 Challenge

In this challenge, each team competes with the other on the recognition accuracy of cooking activities. This section introduces the challenge goal, the dataset, and the evaluation criteria.

2.1 Challenge Goal

The goal of the Cooking Activity Recognition Challenge is to recognize both the macro activity (recipe) and the micro activities taking place in a 30 s window based on acceleration data and motion capture data. The training dataset contains data about 3 subjects and contains all activity labels. The test dataset contains data about the other subject and is not labeled. Participants must submit their predicted macro- and micro activities on the test dataset using their models.

2.2 Dataset

This section introduces the dataset used for this challenge. For more details, please refer to these articles [6–8].

2.2.1 Sensors and Subjects

The data has been collected from four subjects who had attached two smartphones on the right arm and left hip, two smart watches on both wrists, and one motion capture system with 29 markers. The subjects cooked three recipes (sandwich, fruit salad, and cereal) five times each by following a script for each recipe, but acted as naturally as possible.

2.2.2 Data Structure

Training data contains data from three subjects (subject s1, 2, and 3) out of the four subjects, and test data contains the data from the fourth subject (subject 4). Each recording has been segmented into 30 s segments. Each segment was assigned a random identifier, so the order of the segments is unknown. Each sensor data segment is stored in a separate file with the segment-id used to identify related files. Segments of the four sensors at the same time frame were assigned the same identifier.

Groundtruth for all the segments are stored in one file. This file contains one row per file, and each row contains the file name, the macro activity, and the micro activities, all separated by commas; for example, `[subject1_file_939,fruitsalad,Take,Peel,]`, which means that in segment 939, the subject 1 took something and peeled something while making the fruit salad. The micro activity is a multi-label recognition task. The macro activity is of three classes: sandwich, fruitsalad, and cereal; and the micro activity is of ten classes: Cut, Peel, Open, Take, Put, Pour, Wash, Add, Mix, and other.

2.2.3 Statistics

Table 1 shows the number of segments for each subject, the number of annotated classes of macro activity (one in this challenge), max, mean, and min number of annotated classes of micro activities, max, mean, and min length of the segments.

2.3 Evaluation Criteria

Submissions will be evaluated by the average of the accuracy of macro-activity classification (ma) and the average accuracy of micro-activity classification (mi). That is $\frac{ma+mi}{2}$. The average accuracy of micro-activity classification is based on the multi-label accuracy formula. The accuracy of one sample is given by $accuracy = \frac{P \cap G}{P \cup G}$: the number of correct labels predicted (logical product of prediction set P and groundtruth set G) divided by the number of total true and predicted labels (logical sum of P and G).

Table 1 Statistics of the dataset

Subject	Body part	# of segments	# of macro	# of micro			Length		
				Max	Mean	Min	Max	Mean	Min
1	Left hip	80	1	5	2.09	1	159	131.9	1
	Left wrist						8191	2945	0
	Right arm						1470	1309	8
	Right wrist						8257	4484	0
2	Left hip	105	1	6	2.26	1	505	428.3	10
	Left wrist						5986	2171	0
	Right arm						1500	1272	8
	Right wrist						2992	2465	0
3	Left hip	103	1	6	2.30	1	519	429.3	32
	Left wrist						5529	774.6	0
	Right arm						1594	1182	164
	Right wrist						5938	3559	0
4	Left hip	180	1	Unknown	Unknown	Unknown	534	406.7	46
	Left wrist						7143	1126	0
	Right arm						1479	1233	86
	Right wrist						8761	2080	0

3 Method

This section describes the preprocessing to obtain the features from the raw data, the structure of the model, the loss function and the optimizer, and the process of obtaining the activity labels from the predictions obtained by the one-hot vector. Note that our method does not use motion capture data.

3.1 Preprocessing

Handcrafted feature values are extracted from the raw data $[[x_1, \ldots, x_N], [y_1, \ldots, y_N], [z_1, \ldots, z_N]]$, where x, y, z are raw data of x, y, z axes, and N is the number of samples in a 30 s segment. The features are mean, variance, max, min, root mean square (RMS), interquartile range (IQR), and zero crossing rate (ZCR) for x, y, and z axes, respectively. ~~These features are calculated over a 50 ms window slid in steps of 3 s.~~ From the preprocessing, 7 features × 3 axes = 21 dimension feature time series are obtained for one sensor. The dataset includes the data obtained at four body parts, therefore, this process is conducted for each sensor.

3.2 Model

Figure 1 shows the structure of our model. The 21-dimensional feature time-series data is fed into our model consisting of 1D convolutional layer, LSTM layer, linear layer, and sigmold layer. The process at each layer is as follows:

- **1D convolutional layer** has an input of sequence length $N' \times 21$ channels and an output of sequence length $N'' \times$ map size M. N' is the length of the time series after the feature extraction, which is smaller than the original raw data. N'' is the length of the time series after the convolution, which is $N' - K + 1$, where K

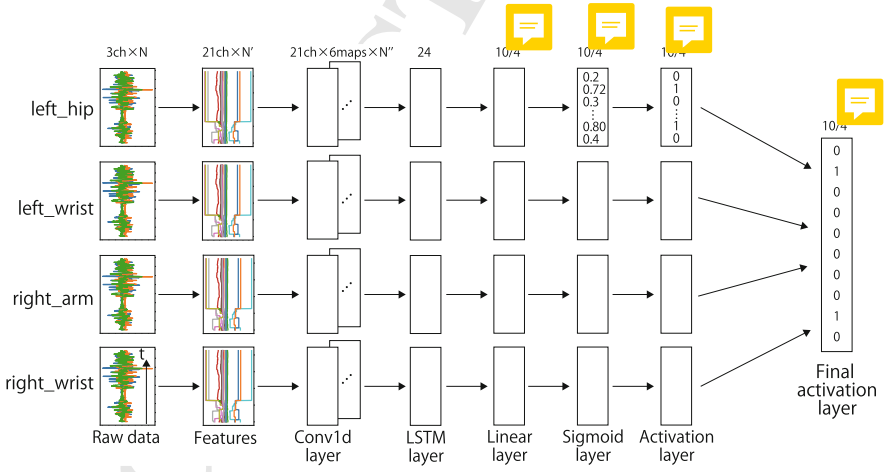

Fig. 1 ~~Our model.~~ The first layer is raw data provided as it is. The second layer is handcrafted features consisting of 21 channels. Conv1d layer is a one-dimensional convolutional layer consisting of 6 maps for 21 channels, 126 maps in total. The 126-channel time-series data will be fed into the next LSTM layer consisting of 24 hidden layers. Then 24-dimensional tensor is shrunk to a 10-dimensional tensor, then the sigmoid function is applied, one-hot encoded with predetermined threshold. At last, four one-hot encoded vectors are merged and the final multi-label prediction is obtained

502437_1_En_8_Chapter ☑ TYPESET ☐ DISK ☐ LE ☑ CP Disp.:16/9/2020 Pages: xxx Layout: T1-Standard

is the kernel size. N' and N'' are variable lengths because the dataset has deficit values and sampling frequency is different for the sensors. Segments whose length is less than 10 are discarded and not fed into the model. Kernel size K is set to 10. Map size M is the number of filters and is set to $6 \times 21 = 126$. Here, the convolution is depth-wise, i.e., the convolution is conducted for each channel and there are 6 filters for each channel.

- **LSTM layer** has an input of sequence length $\times$ 126 channels and an output of a 24-dimensional tensors. The LSTM is many to one. The number of hidden layers is 24, and the last output of the LSTM layers are obtained. At this moment, the output is no longer time series, but one tensor.
- **Linear layer** has an input of a 24-dimensional flattened tensor and an output of ~~10/4~~ dimensional tensors. For macro (recipe) recognition, the output is ~~4~~ dimensions, for micro activities, output is 10 dimensions.
- **Sigmoid layer** applies the sigmoid activation function to the ~~10/4~~ dimensional tensors, which represents the likelihood of the classes.
- ~~**Activation layer** has an 10/4 dimensional tensor and an output of 10/4 dimensional one-hot vector.~~ This layer activates the prediction classes whose values are more than the threshold Th. ~~For micro-activity recognition, the~~ output one-hot vector has multiple 1 elements since the data is multi-labeled, e.g.., [0, 1, 0, 0, 0, 0, 0, 0, 0, 0] or [0, 0, 1, 0, 1, 0, 0, 0, 0, 0]. The threshold Th is determined in the training phase by finding the best accuracy by changing the threshold from 0 to 1. ~~For recipe recognition, the vectors in the sigmoid layer are used, not one-hot encoded.~~

3.3 Loss Function and Optimizer

The models for four sensors are trained separately. The model is trained on BCEWithLogistsLoss in PyTorch for micro activities, and its weight was set to one for all classes. For macro activity, CrossEntropyLoss in PyTorch is used as the loss function. Adam was used for an optimizer for macro- and micro activities.

3.4 Final Prediction Classes Activation

Through the process above, up to four predictions are obtained. At last, our method merges the predictions and outputs the final prediction. In detail, for micro-activity recognition, the four one-hot vectors are summed up; then the final prediction is done as follows. Note that segments whose length is less than 10 are not fed into the system and do not output prediction, therefore, the cases when the number of predictions are one, two, and three are also considered. The number of predictions is one or two, i.e., segments of three or two sensors are too short to be fed into the system; index which is greater than or equal to 1 is activated as the final prediction.

The number of predictions is three or four; index which is greater than or equal to 2 is activated as the final prediction.

For example, suppose that micro activities [“Cut”, “Peel”, “Open”, “Take”, “Put”, “Pour”, “Wash”, “Add”, “Mix”, “other”] are one-hot encoded and predictions of the four sensors are [1, 0, 0, 0, 0, 0, 0, 0, 0, 0] for left hip, [1, 0, 1, 0, 0, 0, 0, 0, 0, 0] for left wrist, [0, 0, 1, 0, 0, 0, 0, 0, 0, 0] for right arm, and [0, 1, 0, 0, 0, 0, 0, 0, 0, 0] for right wrist. The summed-up one-hot vectors is [2, 1, 2, 0, 0, 0, 0, 0, 0, 0] and the number of predictions is four in this case. Indices whose values are greater than or equal to 2, i.e., index 0 and 2, are activated, and our method outputs Cut and Open as a prediction of micro activities for the segment. Index 1 (Peel) is not activated.

For macro-activity recognition, the four vectors in the sigmoid layer are summed up, then the index showing the maximal value is activated since macro activity is a single label. For example, suppose macro activities are [“sandwich”, “fruitsalad”, “cereal”] and the four vectors in the sigmoid layer are [0.1, 0.5, 0.9] for left hip, [0.1, 0.2, 0.6] for left wrist, [0.1, 0.6, 0.8] for right arm, and [0.3, 0.2, 0.7] for right wrist. The summed-up vector is [0.6, 1.5, 3.0]. Index 2, which is showing the greatest value, is activated and our method outputs cereal as a prediction of macro activity for the segment. Threshold is not used for macro-activity recognition since macro activity is a single label.

4 Evaluation

This section describes the evaluation environment, the loss and accuracy in training phase, and processing time in training and testing phases.

4.1 Environment

We implemented the program in Python 3.6.7, PyTorch 1.4.0, CUDA 10.0, and cuDNN 7402. The specification of the computer used for the evaluation is as follows: OS is Windows 10 Pro; CPU is Intel Core i7-8700K 3.7KHz; RAM is DDR4 64GB; and GPU is NVIDIA GeForce RTX 2080Ti GDDR6 11GB. All the data were stored on a local HDD. In the training phase, all the data of subjects 1, 2, and 3 (288 segments) were used for training in one epoch, which iterated 1,000 epochs.

4.2 Result

Table 2 shows the maximum accuracy and minimum loss of micro- and macro activities over 1,000 epochs for the four sensor positions by changing training data and test

Table 2 Maximum accuracy and minimum loss of micro- and macro activities over 1,000 epochs for four sensor positions by changing training data and test data

Activity type	Train data	Test data	Sensor position	Max. accuracy	Min. loss
Micro	Subjects 1, 2	Subject 3	Left hip	0.593	0.396
			Left wrist	0.556	0.454
			Right arm	0.591	0.394
			Right wrist	0.405	0.498
	Subjects 2, 3	Subject 1	Left hip	0.597	0.370
			Left wrist	0.432	0.536
			Right arm	0.516	0.393
			Right wrist	0.441	0.479
	Subjects 1, 3	Subject 2	Left hip	0.564	0.381
			Left wrist	0.534	0.490
			Right arm	0.596	0.374
			Right wrist	0.432	0.452
	Subjects 1, 2, 3	Subjects 1, 2, 3	Left hip	0.717	0.260
			Left wrist	0.761	0.245
			Right arm	0.769	0.208
			Right wrist	0.688	0.261
Macro	Subjects 1, 2	Subject 3	Left hip	0.520	1.051
			Left wrist	0.519	1.008
			Right arm	0.539	1.078
			Right wrist	0.510	1.090
	Subjects 2, 3	Subject 1	Left hip	0.494	1.097
			Left wrist	0.298	1.108
			Right arm	0.603	1.029
			Right wrist	0.268	1.140
	Subjects 1, 3	Subject 2	Left hip	0.535	1.062
			Left wrist	0.596	1.008
			Right arm	0.520	1.047
			Right wrist	0.489	1.081
	Subjects 1, 2, 3	Subjects 1, 2, 3	Left hip	0.904	0.280
			Left wrist	0.905	0.366
			Right arm	0.911	0.271
			Right wrist	0.992	0.065

data. ~~The accuracy was calculated using the one-hot vectors in the activation layer in Fig. 1.~~ The loss was calculated using the vectors in the sigmoid layer in Fig. 1.

From these results, the average accuracy of 0.521 and 0.491 were achieved among subjects 1, 2, and 3 in leave-one-subject-out manner for micro- and macro activities,

Table 3 CPU and GPU memory usage and time taken in training and testing. These figures are when data of four body parts are processed at once

Resource	Macro	Micro
CPU memory	2391 MB	2391 MB
GPU memory	1.6 GB	1.6 GB
Training time (1,000 epoch)	21.554 s	28.891 s
Testing time (1,000 epoch)	58.042 s	59.299 s

respectively. Considering ten multi-label micro activities, it would be said that 0.521 accuracy is good, while 0.491 accuracy for 3-class macro activity can be improved. Comparing the results using test data of subjects 1, 2 and 3, there seems to be no difference among them, showing that the data of the three subjects are similar. However, the results of both wrists are lower than the hip and the arm. It can be said that individual characteristics appeared in hand movements while cooking and the constructed model cannot be generalized.

When data of subjects 1, 2, and 3 are used for both training and testing, accuracy for micro- and macro activities were improved to 0.734 and 0.928, respectively. Note that for submitted results for the data of subject 4, our model was trained separately for the body parts with the data of subjects 1, 2, and 3, and the model at 1,000th epoch was used for testing the data of subject 4. Table 3 shows the memory usage on CPU and GPU, and processing time taken in the training phase and the testing phase.

5 Conclusion

This paper reported the solution of our team "Rits's cooking" to Cooking Activity Recognition Challenge held at International Conference on Activity and Behavior Computing (ABC2020). Our approach leverages the convolutional layer and LSTM to recognize macro activities (recipe), and micro activities (body motion). The evaluation results showed that the average accuracy of 0.521 and 0.491 were achieved among subjects 1, 2, and 3 in leave-one-subject-out manner for micro- and macro activities, respectively. We plan to construct the streamline model without handcrafted features and majority vote.

6 Appendix

6.1 Used Sensor Modalities

Four acceleration sensors at the left hip, left wrist, right arm, and right wrist from three subjects were used. Mocap data was NOT used.

6.2 Features Used

Seven kinds of features were used: Mean, variance, max, min, root mean square (RMS), interquartile range (IQR), and zero crossing rate (ZCR). These features are extracted for x, y, and z axes, respectively.

6.3 Programming Language and Libraries Used

Python 3.6.7 was used. For network implementation, PyTorch 1.4.0 was used.

6.4 Window Size and Post-processing

~~Window size is 500 ms and step size is one sample.~~

6.5 Training and Testing Time

Training time (1,000 epoch) was 21.554 s for macro activity and 28.891 s for micro activity. Testing time (1,000 epoch) was 58.042 s for macro activity and 59.299 s for micro activity.

6.6 Machine Specification

OS: Windows 10 Pro. CPU: Intel Core i7-8700K 3.7KHz. RAM: DDR4 64GB. GPU: NVIDIA GeForce RTX 2080Ti GDDR6 11GB.

References

1. Bayat, A., Pomplun, M., Tran, D.A.: A study on human activity recognition using accelerometer data from smartphones **34**, 450–457 (2014)
2. Wang, A., Chen, G., Yang, J., Zhao, S., Chang, C.: A comparative study on human activity recognition using inertial sensors in a smartphone **16**(11), 4566–4578 (2016)
3. Javier Ordóñez, F., Roggen, D.: Deep, convolutional and lstm recurrent neural networks for multimodal wearable activity recognition **16**, 1–25 (2016)
4. Atallah, L., Lo, B., King, R., Yang, G.: Sensor positioning for activity recognition using wearable accelerometers **5**(4), 320–329 (2011)
5. Casale, P., Pujol, O., Radeva, P.: Human activity recognition from accelerometer data using a wearable device. In: Pattern Recognition and Image Analysis, pp. 289–296 (2011)
6. Lago, P., Takeda, S., Adachi, K., Shamma Alia, S., Matsuki, M., Benaissa, B., Inoue, S., Charpillet, F.: Cooking activity dataset with macro and micro activities (2020). https://doi.org/10.21227/hyzg-9m49
7. Lago, P., Takeda, S., Shamma Alia, S., Adachi, K., Benaissa, B., Charpillet, F., Inoue, S.: A dataset for complex activity recognition with micro and macro activities in a cooking scenario (2020)
8. Shamma Alia, S., Lago, P., Takeda, S., Adachi, K., Benaissa, B., Ahad, Md A.R., Inoue, S.: Summary of the cooking activity recognition challenge (2020)
9. Jiang, W., Yin, Z.: Human activity recognition using wearable sensors by deep convolutional neural networks. In: Proceedings of the 23rd ACM International Conference on Multimedia, pp. 1307–1310 (2015)
10. Wang, W., Liu, A.X., Shahzad, M., Ling, K., Lu, S.: Device-free human activity recognition using commercial wifi devices **35**(5), 1118–1131 (2017)
11. Chen, Y., Shen, C.: Performance analysis of smartphone-sensor behavior for human activity recognition **5**, 3095–3110 (2017)
12. Chen, Y., Zhong, K., Zhang, J., Sun, Q., Zhao, X.: LSTM Networks for Mobile Human Activity Recognition. In: 2016 International Conference on Artificial Intelligence: Technologies and Applications, pp. 50–53 (2016)
13. Chen, Y., Xue, Y.: A deep learning approach to human activity recognition based on single accelerometer. In: 2015 IEEE International Conference on Systems, Man, and Cybernetics, pp. 1488–1492 (2015)

Author Queries

Chapter 8

Query Refs.	Details Required	Author's response
AQ1	In the sentence "For micro activities ...", please check for "vote by the body parts" and correct if necessary.	
AQ2	Please check the sentences starting from "Note that segments..." up to the end of the para for clarity.	
AQ3	Please check for the usage of "single label" in the para starting with the sentence "For macro-activity recognition...".	

MARKED PROOF

Please correct and return this set

Please use the proof correction marks shown below for all alterations and corrections. If you wish to return your proof by fax you should ensure that all amendments are written clearly in dark ink and are made well within the page margins.

Instruction to printer	*Textual mark*	*Marginal mark*
Leave unchanged	··· under matter to remain	New matter followed by
Insert in text the matter indicated in the margin		or (x2)
Delete	through single character, rule or underline or through all characters to be deleted	or (x2)
Substitute character or substitute part of one or more word(s)	through letter or through characters	new character or new characters
Change to italics	under matter to be changed	
Change to capitals	under matter to be changed	
Change to small capitals	under matter to be changed	
Change to bold type	under matter to be changed	
Change to bold italic	under matter to be changed	
Change to lower case	Encircle matter to be changed	
Change italic to upright type	(As above)	
Change bold to non-bold type	(As above)	
Insert 'superior' character	through character or where required	or under character e.g. or
Insert 'inferior' character	(As above)	over character e.g.
Insert full stop	(As above)	
Insert comma	(As above)	
Insert single quotation marks	(As above)	or and/or or
Insert double quotation marks	(As above)	or and/or or
Insert hyphen	(As above)	
Start new paragraph		
No new paragraph		
Transpose		
Close up	linking characters	
Insert or substitute space between characters or words	through character or where required	
Reduce space between characters or words	between characters or words affected	