Appendix

More Features of Chinese Characters

Although the number of Chinese characters is large, the number of commonly used Chinese characters is limited. According to Zhou (1992), 1,000 commonly used characters already cover 90% of Chinese character usage in different corpora, and 2,400 Chinese characters have 99% coverage. Chinese students learn about 2500 characters in primary school.

The stroke order of writing Chinese characters in this study is prescriptive. We do not provide a formal generative model to describe the process, although there could be alternative ways to generate the same character.

Structural Patterns The structural patterns of Chinese characters include single structure, left-right structure, top-down structure, enclosing structure, and special structure. Refer to the National Chinese Character Holographic Resource Application System for knowledge and features of Chinese characters at `http://qxk.ywky.edu.cn/#/`.

- Single structure: it is illustrated as a single character that cannot be decomposed further. E.g. 大由人.
- Left-right structure: it includes left-right structure and left-middle-right structure. E.g. 凯招炒, 街掰辩.
- Top-down structure: it includes top-down structure and top-middle-down structure. E.g. 是药想, 曼率器.
- Enclosing structure: it includes two-sided, three-sided and four-sided enclosures. E.g. 厄这勉, 用同风, 国园回.
- Special structure: it includes frame structure, crossing structure, etc. E.g. 承乘爽.

Primitive Components The number of primitive components varies in different research works, depending on ways of decomposing characters and identifying primitive components. The principles of decomposing in *Specification of Common Modern Chinese Character Components and Component Names* (Ministry of Education of the People's Republic of China, 2009) are based on character building methods, visual shape, systematic consistency, and practical applications.

There are several practical decomposing rules. (1) If the etymological origin or semantic structure of characters are clear, then decompose them. For example, "分" (split) can be decomposed into "八" (visually two parts) and "刀" (knife). If not, then decompose characters according to the visual shapes. For example, "朋" can be decomposed into "月" and "月". (2) Components with crossing or overlapping strokes will not be decomposed. For example, "串" cannot be decomposed into "中" and "中"; "东" cannot be decomposed into "七" and "小". (3) If all parts of a character or a component are bound components or no longer constitute other Chinese characters, then decomposing will not apply. For example, "非" cannot be decomposed into two parts. (4) If the parts of one component are separated because of the composition of the character, the split parts will be united and keep the original shape of the parts. For example, "裹" should be decomposed into "衣" and "果", and two parts of "衣" should not be treated as individual components.

CChar Dataset

Image Collection Images in CChar are derived from open-source font files. Data cleaning includes aligning frames of characters with stroke sequences and converting formats.

Annotation The annotation task is to recognize spatial patterns of Chinese characters and decompose them into smaller components by labeling paired brackets "[]", according to stroke sequences. For each smaller component, decompose until it can no longer be split into smaller parts further. For example, given the Chinese character "仁" and its stroke sequence "3211", the task is to recognize the "left-right" pattern, add pair brackets to components "亻" and "二", and confirm these components cannot be decomposed further. The procedures of annotation follow the Figure 4.

The parsing of components in Chinese characters will stop when only when all left components are primitive components that cannot be decomposed further. Similar to syntactic parsing, there are ambiguities. The identification of primitive components here reflects more linguistic preference rather than visual preference. Here are practical guidelines:

- Single structure: single characters no longer need to be split. Refer to the list of primitive components when necessary.
- Left-right structure and top-down structure: for the characters with three parts, if two of them could form a larger component, then adopt nested binary parsing. Otherwise, consider a left-middle-right structure or a top-middle-bottom structure.
- Special structure: for the characters with frame structures or top-middle-bottom structures, check whether the separated and discontinued parts could form a complete component, and wrap the entire component in brackets when necessary.

The annotation was completed by an experienced Chinese language teacher who has taught writing Chinese characters in primary school for over 20 years. To guarantee characters are decomposed in a consistent and systematic way, we also check the consistency of components in the list of primitive components with programs and human inspection. It takes 4 weeks to annotate and 4 weeks to validate the consistency.

Descriptive Statistics The average length of stroke sequences without hierarchical annotations is 10.69. The length in CChar follows a normal distribution (see Figure 6). The character with the longest stroke sequence in CChar is "爨" (cooking), and its length is 30 (see its interpretation in Figure

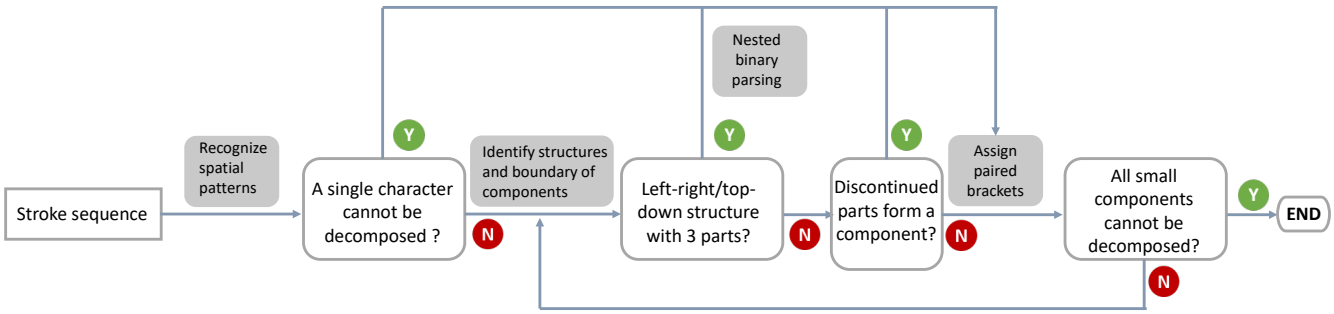

Figure 4: The procedures of annotation.

Label	Percentage	Label	Percentage
1	16.35	1	30.43
2	10.25	2	19.09
3	8.58	3	15.98
4	9.01	4	16.77
5	9.53	5	17.73
"["	23.14		
"]"	23.14		

Table 6: The distribution of labels in stroke sequences with and without brackets.

5). The distribution of stroke and bracket labels in stroke sequences is shown in Table 6. There are 549 primitive components in CChar. The average length of primitive components is 4.7. The distribution of their length is shown in Figure 7. The number of hierarchical levels in CChar is from 1 to 6 (see Table 7). There are 260 characters with discontinuous structures, which account for 3.8% of CChar characters.

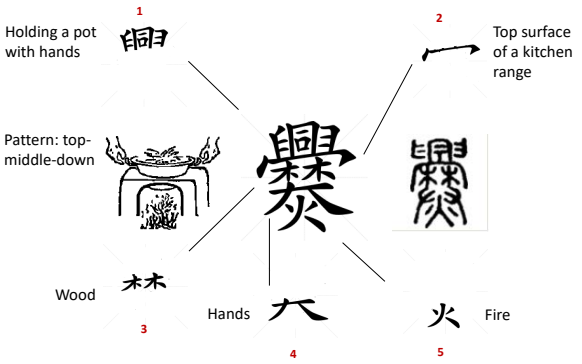

Figure 5: The interpretation of "爨" (cooking).

Experiments

Hyperparameters of Models Detailed hyperparameters of GRU, LSTM and Transformers in different experiments are illustrated in Table 8, Table 9, and Table 10.

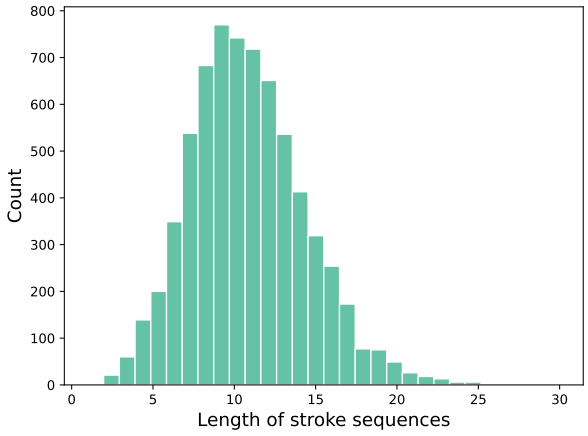

Figure 6: Length of stroke sequences in CChar.

Level	Train	Val	Test	Sum
1	238	48	60	346
2	1729	370	387	2486
3	2123	477	429	3029
4	644	117	138	899
5	54	12	10	76
6	0	2	2	4
sum	4788	1026	1026	6840

Table 7: The distribution of hierarchical levels.

Experiment 1 The visualization of attention in 8 heads shows Transformers could notice the boundaries of larger local components from different profiles, no matter the target sequence is annotated with hierarchies or not (see Figure 8 and Figure 9). By comparing Figure 8 and Figure 9, the results suggest that it is easier for Transformer models to recognize components and their boundaries when there are brackets as explicit cues. Specifically, Transformer models could capture the dependency between paired brackets at the beginning and the end of sequences (see the sub-image at row 2, column 2 in Figure 8).

The errors in predicted sequences with hierarchies show

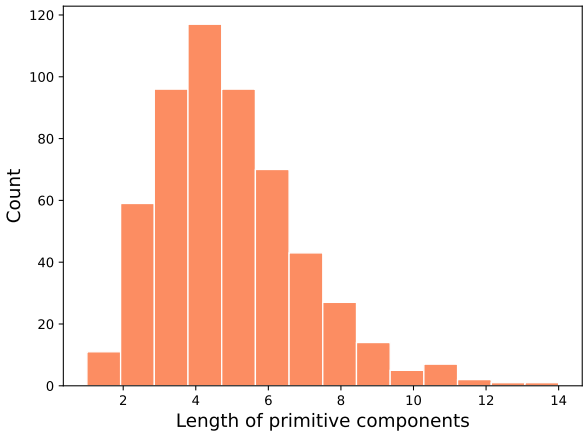

Figure 7: Length of primitive components in CChar.

Hyperparameter	GRU	LSTM	Transformer
dimension of features	512	512	512
encoder layer	2	2	2
decoder layer	2	2	2
hidden size	512	512	512
attention head	-	-	8
drop out	0.5	0.5	0.1
epochs	50	50	150

Table 8: Hyperparameter setting in experiment 1.

very diverse and irregular patterns. It is not easy to recover tree structures, build one-to-one alignment at multiple levels, and classify the errors systematically. Here, we select the predicted sequences that cannot totally match with true sequences, and count the cases when the brackets are not paired for each model. In Table 11, the results show that Transformer models make very few mistakes by predicting brackets with odd numbers, which implies Transformer models are better at learning long dependency.

Experiment 3 We control the depth of hierarchies and make data points with different hierarchies are equal in 20 different subgroups. Since the number of characters with 5 or 6 levels is much smaller (see Table 7), we only consider the data with hierarchies from 1 to 4. The distribution in different subgroups is illustrated in Table 12. Simple linear regression is adopted to test if the depth of hierarchical structures (X) significantly predicts the Levenshtein ratios (Y) of different models.

- GRU: The fitted regression model is $\hat{Y} = -0.0203X + 0.7519$. The overall regression is statistically significant ($R^2 = 0.425$, $F(1,78) = 57.64$, $p < .000$). The depth of hierarchies is found to be a significant predictor of Levenshtein ratios ($p < .000$).

- LSTM: The fitted regression model is $\hat{Y} = -0.0193X + 0.7667$. The overall regression is statistically significant ($R^2 = 0.469$, $F(1,78) = 68.85$, $p < .000$). It is found that the depth of hierarchies significantly predict Levenshtein ratios ($p < .000$).

- Transformer: The fitted regression model is $\hat{Y} = 0.0085X + 0.6521$. The overall regression is statistically significant ($R^2 = 0.202$, $F(1,78) = 19.73$, $p < .000$). It is discovered that the depth of hierarchies is a significant predictor of Levenshtein ratios ($p < .000$).

Hyperparameter	GRU	LSTM	Transformer
dimension of features	256	256	256
encoder layer	2	2	2
decoder layer	2	2	2
hidden size	256	256	256
attention head	-	-	4
drop out	0.5	0.5	0.1
epochs	50	50	150

Table 9: Hyperparameter setting in experiment 2.

Hyperparameter	GRU	LSTM	Transformer
dimension of features	256	256	256
encoder layer	1	1	1
decoder layer	1	1	1
hidden size	256	256	256
attention head	-	-	4
drop out	0.5	0.5	0.1
epochs	30	30	50

Table 10: Hyperparameter setting in experiment 3.

Model	Unpaired brackets (%)
GRU	61.05
LSTM	52.48
Transformer	5.00

Table 11: The percentage of unpaired brackets in total errors for each model.

level	train	val	test	sum
1	238	49	60	347
2	238	49	60	347
3	238	49	60	347
4	238	49	60	347
sum	952	196	240	1388

Table 12: The controlled distribution of hierarchical levels in subsets.

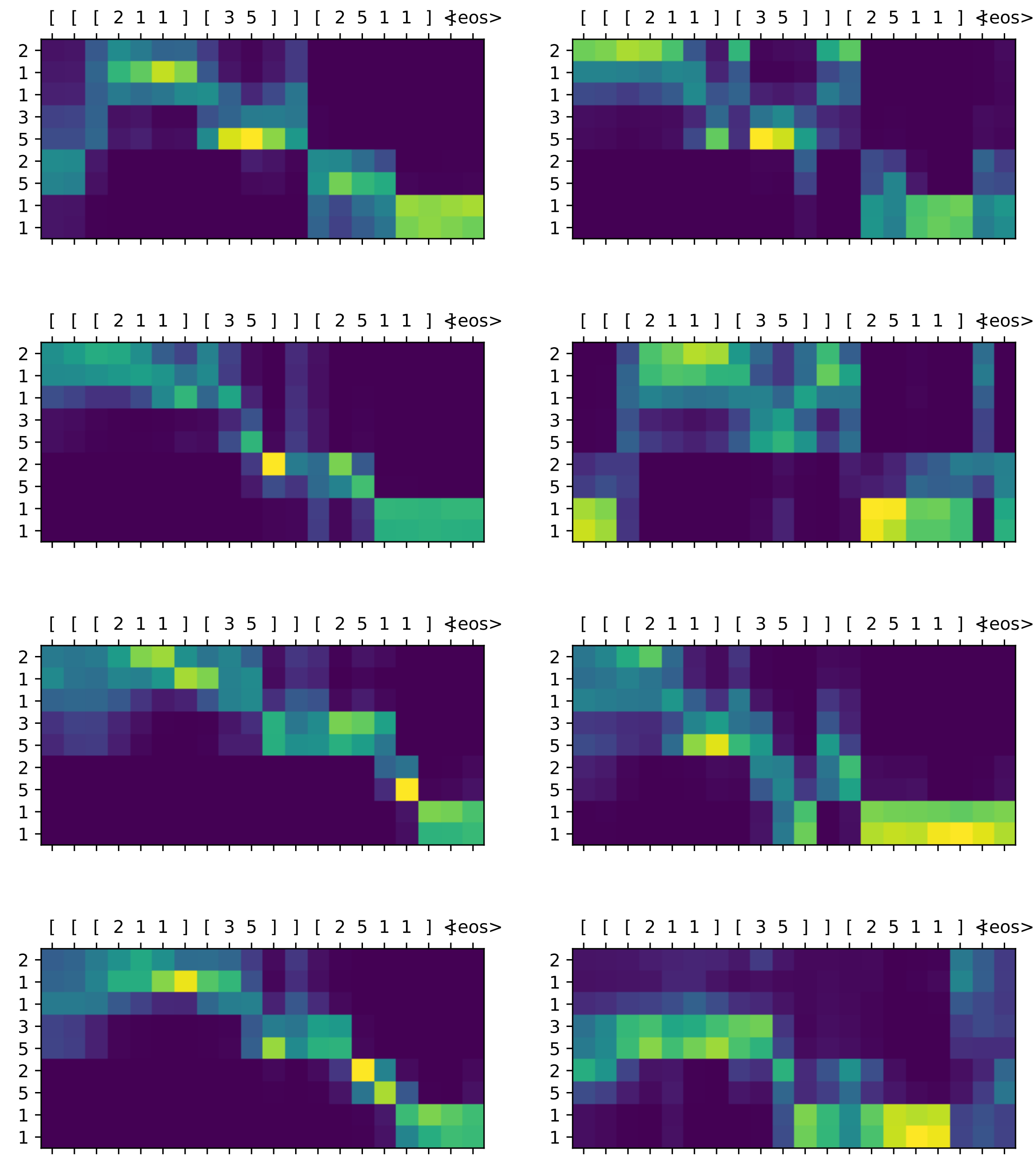

Figure 8: The attention of Transformers in predicting “背” (back) with hierarchies. The true sequence is “[[[211][35]][2511]]”.

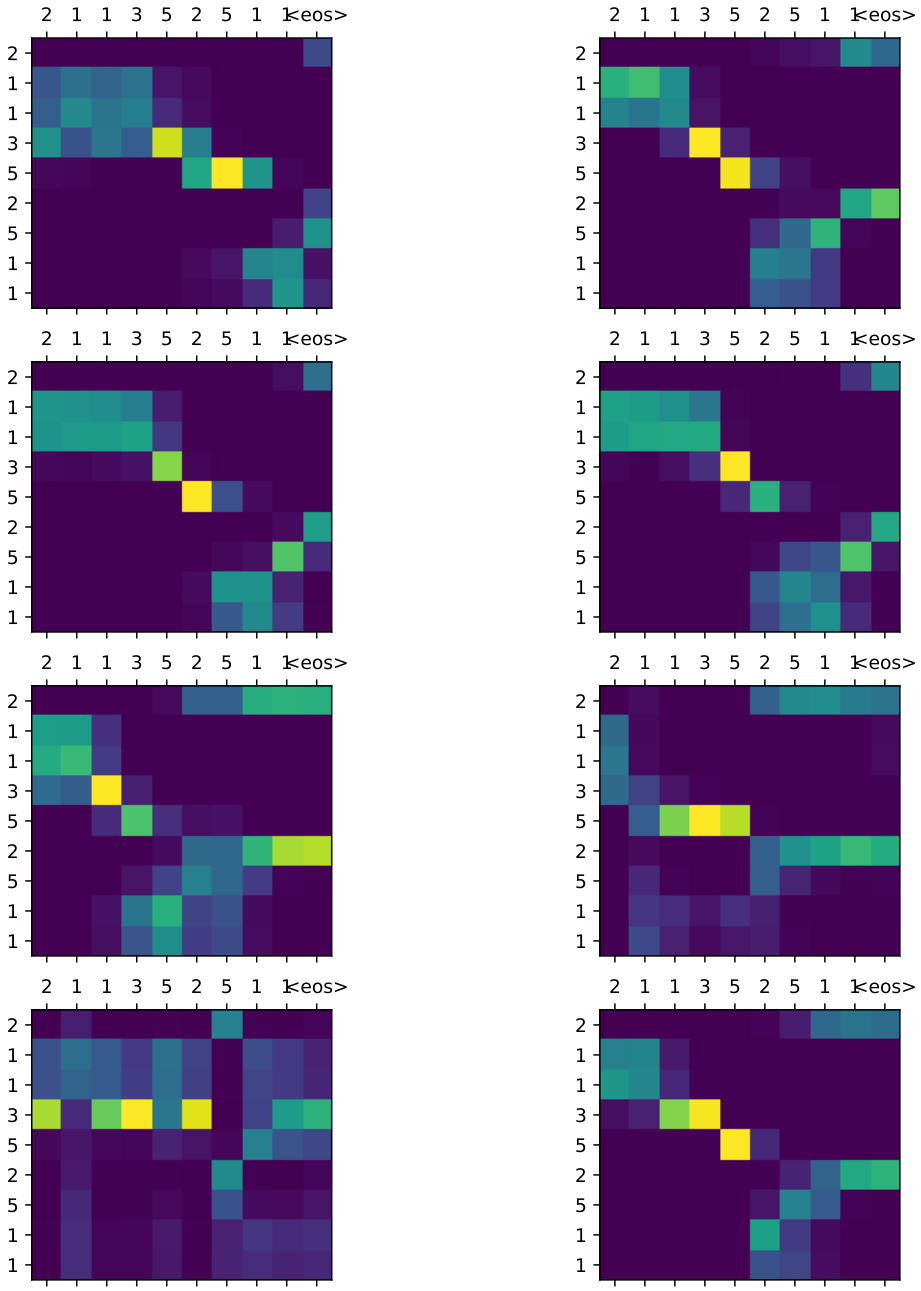

Figure 9: The attention of Transformers in predicting “背” (back) without hierarchies. The true sequence is “211352511”.