Learning Morphological Patterns with Neural Networks

Xiaomeng Ma

CUNY Graduate Center

Introduction

- English verb inflections have rule-governed items and exceptions: e.g. for V;PST
  - regular rule: + ed
  - double spell rule: prefe**r** - prefe**rr**ed
  - back ablaut rule: s**i**ng - s**a**ng
  - vowel shortening rule: bl**ee**d - bl**e**d, l**ea**d - l**e**d
  - ...
  - exceptions: go - went, do - did
- Learning Premise
  1. neural networks learn from printed data
  2. **Not** human learns from auditory data

Introduction

- Neural Networks can learn English inflection patterns (e.g. Kirov and Cotterell, 2018; Corkery et al., 2019; Calderone et al., 2021). Especially, character level transformer achieved good performance on morphology transductions (Wu et al., 2020).
- Remaining Questions:
  1. What kind(s) of patterns does the model learn?
  2. How much input does the model need to learn a pattern?
- Replication Wu et al. (2020)'s transformer model to answer these questions.

Dataset

- CoNLL-SIGMORPHON Shared Task 2017
  1. root(e.g. tar), inflection(e.g. tarred), type(e.g. V;PST)
  2. training: english-train-high (10,000)
  3. validating: english-dev (1,000)
  4. testing: english-uncovered-text(1,000)
- Added Columns:
  1. diff: Character level string differences: str(root) - str(string)
     e.g. ‘speak’ - ‘speaking’ = ‘-ing’, ‘take’ - ‘took’ = ‘ake-ook’
  2. rule: e.g. ‘double spell’, ‘+s’..
  3. reg: Regular and Irregular
- Example Data

root	**infl**	**type**	**diff**	**rule**	**reg**
show	shows	V;3;GS;PRS	-s	+s	Regular
hug	hugged	V;PST	-ged	double spell	Irregular

Dataset Description

Table 1: Summary of training tokens for each type

Type	Rule	Training tokens	Regular tokens	Regular %
V;3;SG;PRS	+s	2025	1842	91
V;PST	+ed	2016	817	41
V;V.PTCP;PRS	+ing	1959	949	48
V;V.PTCP;PST	+ed	1992	788	40
V;NFIN	∅	2008	2008	100
Total		10,000	6394	64

Data Description

Table 2: Summary of tokens for different rules

	+s	+es	y-ies	special	
V;3;SG;PRS	1842	103	74	6	
	+d	+ed	double spell	y-ied	special
V;PST	856	817	135	79	109
	+ing	-e+ing	double spell	special	
V;V.PTCP;PRS	949	831	166	13	
	+d	+ed	double spell	y-ied	special
V;V.PTCP;PST	862	788	154	66	111
V;NFIN	2008 no change				

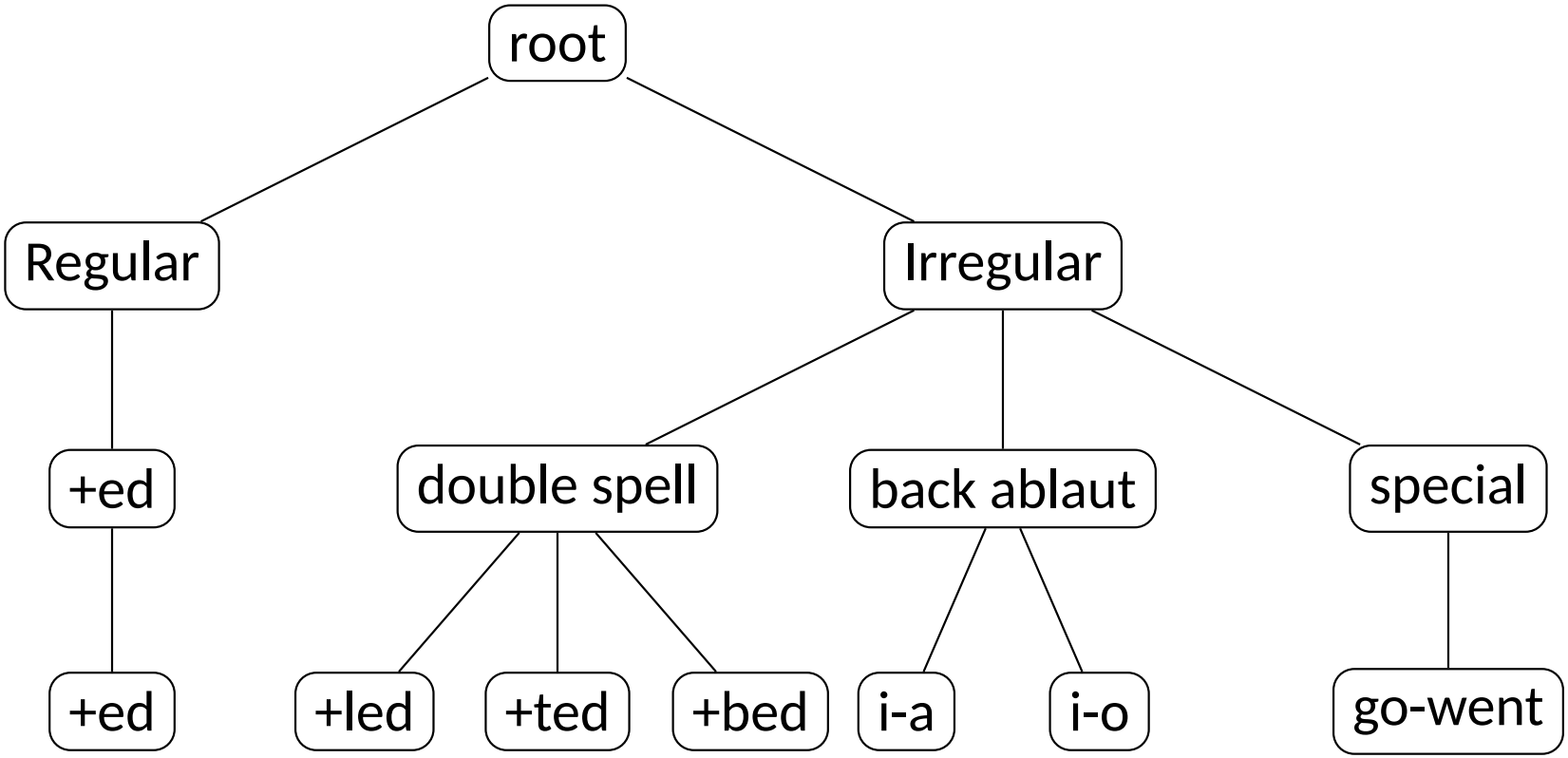
Data Description

Table 3: Summary of character-level differences of different types

Type	Patterns Types	Patterns Token >3	Example Patterns
V;3;SG;PRS	7	3	*-zes*, *-ses*
V;PST	59	30	*-ged*, *-ted*
V;V.PTCP;PRS	20	13	*ie-ying*,*-ling*
V;V.PTCP;PST	62	32	*-en*, *i-u*
V;NFIN	∅		

Patterns

- 3 levels of patterns:
  a) regular; b) rule-based patterns; c) character-level patterns



Past tense inflection as an example. Somewhat like exemplar abstraction process

Experiment 1. Research Questions

- What kind of patterns can the model learn?
  1. character-level patterns:
     - Accuracy and pattern tokens should have a strong positive correlation
     - The model is not able to generalize patterns
  2. rule-based patterns:
     - Are those rules the same as existing linguistic rules?
     - Does the model generate some new rules?
  3. regular vs irregulars:
     - High accuracy on regular items, extremely low accuracy on irregular items
     - Model can only learn the regular rule (e.g. '+ed', '+ing') and won't produce other patterns

Model Details

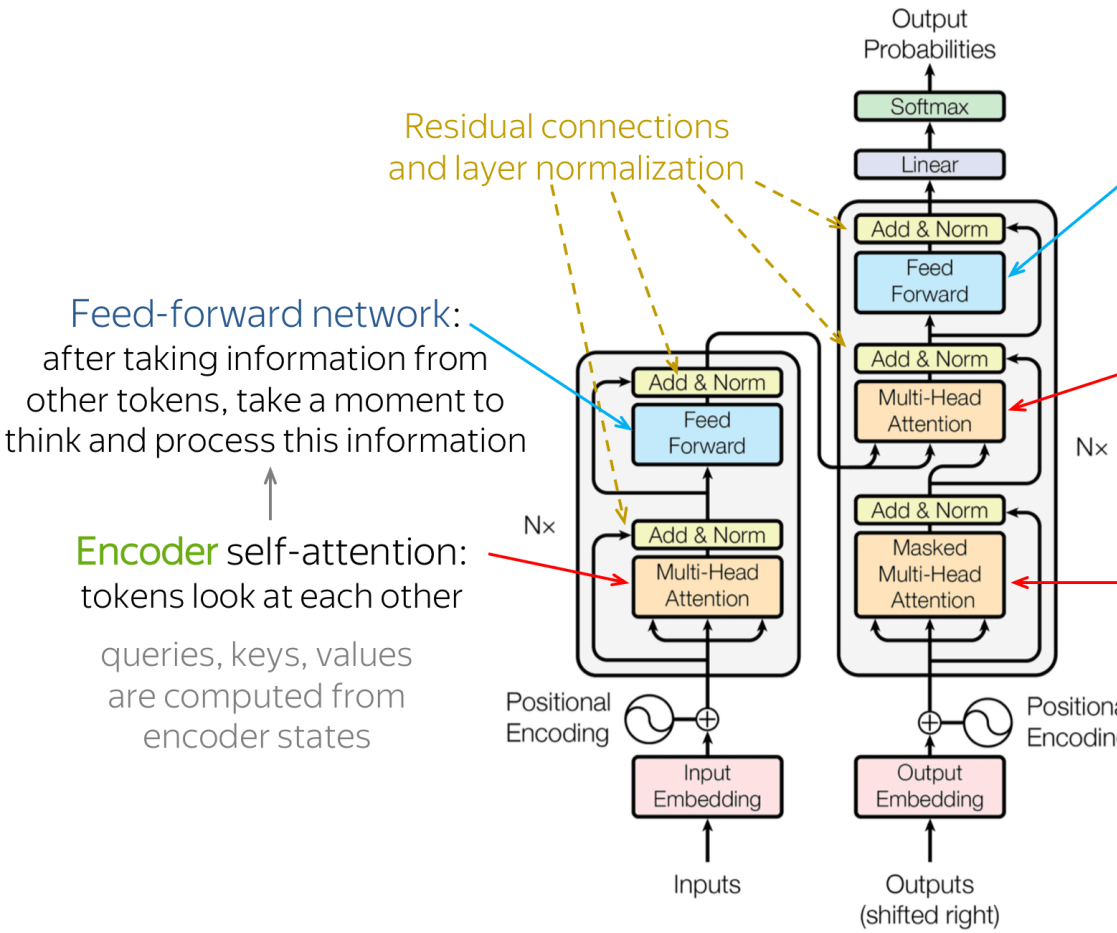
- Model 1. All inflections transformation
  1. Training token: 10,000
  2. Input: root + type,
     e.g. ‘diy+V;3;SG;PRS’, ‘hug+V;PST’
  3. Ouput: inflection,
     e.g. ‘diys’; ‘hugged’
- Model 2. Each inflection’s transformation
  1. Training tokens:$\sim$ 2000 for each inflection
  2. Input: root,
     e.g ‘hug’
  3. Output:inflection, e.g.
     for V;PST: ‘hugged’

Model Details

- Differences between Model 1 and 2:
  1. Model 1 sees some patterns across different tags in the training data, e.g. '+ed' exists for both V;PST and V;V.PTCP;PST
  2. Model 2 only sees the patterns in one type of inflection
- Shared Features:
  1. character-level tokenization
  2. embedding dimension: 32, latent space: 128
  3. attention heads: 4
  4. total parameters: 76,784
- Models are trained on GoogleColab using `TensorFlow`

Model Architecture

Transformer-based Encoder-Decoder model



Residual connections and layer normalization

Feed-forward network: after taking information from other tokens, take a moment to think and process this information

Encoder self-attention: tokens look at each other

queries, keys, values are computed from encoder states

Feed-forward network: after taking information from other tokens, take a moment to think and process this information

Decoder-encoder attention: target token looks at the source

queries – from decoder states; keys and values from encoder states

Decoder self-attention (masked): tokens look at the previous tokens

queries, keys, values are computed from decoder states

picture credit: Lena Voita NLP Course

Results: Accuracy

Table 4: Summary of regular and irregular items accuracy

	Model 1 Accuracy			Model 2 Accuracy		
Type	Total	Regular	Irregular	Total	Regular	Irregular
V;PST	0.91	0.95	0.86	0.87	0.94	0.81
V;V.PTCP;PST	0.84	0.95	0.76	0.83	0.96	0.73
V;V.PTCP;PRS	0.90	0.89	0.91	0.92	0.98	0.86
V;3;SG;PRS	0.95	0.96	0.92	0.88	0.91	0.64
V;NFIN	0.97	0.97	Nan	0.98	0.98	Nan
Overall	0.930	0.995	0.869			

- Model 1 generally has better accuracy in Irregulars
- The relatively high accuracy in irregulars suggest that the model learned more than general level of regularity

Results: Accuracy

Table 5: Summary of 3;SG;PRS accuracy

V;3;SG;PRS		Model 2			Model 1				
rule	diff	✓	✗	Acc	✓	✗	Acc	train tokens	train ratios
+es	-es	10	7	0.59	16	1	0.94	103	5.09%
+s	-s	168	17	0.91	181	4	0.98	1842	90.96%
special	-ses	0	1	0.00	0	1	0	0	0.00%
y-ies	y-ies	6	1	0.86	7	1	0.88	74	3.65%

- Model 1 and 2 share the same patterns (since none of the differences is found in other types of inflection)
- Model 1 has better performance than Model 2
- It looks like the more patterns the model saw, the higher accuracy

Results: Accuracy

Table 6: Summary of V.PTCP;PRS Accuracy

V;V.PTCP;PRS		Model 2			Model 1				
rule	diff	✓	✗	Acc	✓	✗	Acc	train tokens	train ratios
regular	-ing	99	2	0.98	92	9	0.91	949	48.44%
e-ing	e-ing	84	0	1.00	84	0	1.00	831	42.42%
double spell	∅	3	13	0.19	10	6	0.63	155	7.91%
	-bing	1	2	0.33	3	0	1.00	13	0.66%
	-ging	0	1	0.00	1	0	1.00	17	0.87%
	-ling	0	3	0.00	3	0	1.00	29	1.48%
	-ming	0	2	0.00	0	2	0.00	12	0.61%
	-ning	0	1	0.00	0	1	0.00	16	0.82%
	-ping	1	1	0.50	2	0	1.00	30	1.53%
	-ting	1	3	0.25	1	3	0.25	38	1.94%
special	e-ting	0	1	0.00	0	1	0.00	0	0.00%
Spearman Correlation	Acc-ratio	0.72*			0.38				
	✓-token	0.77*			0.81*				
	✗-token	0.05			0.27				

Results: Basic Level Accuracy

V;V.PTCP;PST		Model 2					Model 1				
rule	diff	✓	✗	Acc	each train tokens	ratio	✓	✗	Acc	total train tokens	ratio
regular	-ed	79	3	0.96	788	39.56%	78	4	0.95	1605	16.05%
+d	-d	73	7	0.91	862	43.27%	77	3	0.96	1718	17.18%
y-ied	y-ied	6	0	1.00	66	3.31%	5	1	0.83	145	1.45%
double spell	∅	2	8	0.20	104	5.22%	4	6	0.4	198	1.98%
	-bed	1	1	0.50	9	0.45%	1	1	0.5	16	0.16%
	-fed	0	1	0.00	0	0.00%	0	1	0	0	0.00%
	-led	0	2	0.00	27	1.36%	1	1	0.5	47	0.47%
	-ned	0	1	0.00	8	0.40%	0	1	0	20	0.20%
	-ped	0	1	0.00	28	1.41%	1	0	1	54	0.54%
	-red	1	1	0.50	12	0.60%	1	1	0.5	18	0.18%
	-ted	0	1	0.00	20	1.00%	0	1	0	43	0.43%
d-t	d-t	0	2	0.00	3	0.15%	0	2	0	7	0.07%
+n	-n	0	1	0.00	19	0.95%	0	1	0	19	0.19%
no change	-	0	2	0.00	11	0.55%	0	2	0	2035	20.35%
special	ad-d	0	1	0.00	2	0.10%	0	1	0	4	0.04%
special	eak-oken	0	1	0.00	1	0.05%	0	1	0	1	0.01%
special	ear-orn	0	1	0.00	0	0.00%	0	1	0	0	0.00%
special	ear-orne	0	1	0.00	0	0.00%	0	1	0	0	0.00%
special	ep-pt	0	1	0.00	2	0.10%	0	1	0	2	0.02%
special	ink-ought	0	1	0.00	2	0.10%	0	1	0	4	0.04%
special	y-id	0	1	0.00	4	0.20%	0	1	0	10	0.10%
Correlation	Acc-ratio	0.6**					0.64**				
	✓-token	0.61**					0.67***				
	✗-token	0.34					0.46*				

Results: Accuracy

V;PST		Model 2					Model 1				
rule	diff	✓	✗	Acc	each train token	ratio	✓	✗	Acc	total train token	ratio
regular	-ed	98	6	0.94	817	40.53%	99	5	0.95	1605	16.05%
y-ied	y-ied	8	0	1.00	79	3.92%	8	0	1.00	145	1.45%
+d	-d	86	3	0.97	856	42.46%	89	0	1.00	1718	0.43%
double spell	∅	1	10	0.09	93	4.61%	5	6	0.45	195	17.18%
	-bed	0	1	0.00	7	0.35%	1	0	1.00	16	1.95%
	-ded	1	0	1.00	9	0.45%	1	0	1.00	20	0.16%
	-ged	0	1	0.00	18	0.89%	1	0	1.00	40	0.20%
	-ked	0	1	0.00	4	0.20%	1	0	1.00	9	0.40%
	-led	0	5	0.00	20	0.99%	0	5	0.00	47	0.09%
	-ned	0	1	0.00	12	0.60%	1	0	1.00	20	0.47%
	-ted	0	1	0.00	23	1.14%	0	1	0.00	43	0.20%
no change	-	0	3	0.00	14	0.69%	0	3	0.00	2035	20.35%
special	a-o	0	1	0.00	1	0.05%	0	1	0.00	1	0.01%
special	e-o	0	1	0.00	2	0.10%	0	1	0.00	2	0.02%
special	ed-d	0	1	0.00	7	0.35%	1	0	1.00	8	0.08%
special	ee-aw	0	1	0.00	1	0.05%	0	1	0.00	1	0.01%
special	i-o	0	2	0.00	12	0.60%	0	2	0.00	15	0.15%
special	ind-ound	0	1	0.00	2	0.10%	0	1	0.00	4	0.04%
Correlation	Acc-ratio	0.53*					0.23				
	✓-token	0.60*					0.52*				
	✗-token	0.39					0.21				

Interim Results

- What type of patterns did the model learn?
  1. The learned patterns are more detailed than regular and more general than character-level patterns.
  2. Not all linguistic rule-based patterns learned: e.g. double spell rule that appears in V;V.PTCP;PRS (3/16), V;V.PTCP;PST(2/10), V;PST(1/11)
  3. If not linguistic rules, what rules did model use to generate patterns?
- The relationship between train tokens and accuracy:
  1. Generally, there is a strong positive correlation between accuracy and train ratio and correct tokens with train tokens.
  2. Model is also able to learn with few train tokens, e.g. y-ies/y-ied has high accuracy with only $\sim$ 60-80 train tokens

Error Pattern Analysis

- Evaluating generation errors (Gorman et al., 2019):
  - Error Types
    1. Target errors: consists of cases where the gold data is incorrect
    2. Silly errors: 'bizarre' errors which defy purely linguistic characterization (e.g. 'membled' for 'mailed')
    3. Allomorphy errors: misapplication of existing allomorphic patterns
    4. Spelling errors: forms that do not follow orthographic conventions but are otherwise correct
  - Tasks
    1. Evaluating models: 2 Encoder-Decoder models: A.`UE-LMU-I` and B.`CLUZH-7` (top 2 systems)
    2. Dataset: 52 languages (including English) in CoNLLL-SIGMORPHON 2017 Shared Task dataset
  - Results for English

Error	Target	Silly		Allomorphy		Spelling	
Model		A	B	A	B	A	B
English	3	0	0	18	18	7	11

Error Pattern Analysis

- Spelling errors: 0
- Target errors: 3
- Silly errors:

pattern	type	M.1	M.2	Example
‘eeeee/iiiii’	V;V.PTCP;PST, V;PST,	7	10	weteeeeee
altering letter	V;V.PTCP;PST V.PST V;3;SG;PRS	2	6	rrquisitions
total		9	16	

Error Analysis

- Allormorphy errors:

pattern	type	M.1	M.2	Example
no change	V;V.PTCP;PST; V;PST; V;3;SG;PRS	2	13	opt, dis
y-i	V;V.PTCP;PST; V;PST	1	5	overjoied; stereotiped
+ed	V;V.PTCP;PST	2	2	miscomed
+d	V;PST	3	1	enwrited
+s	V;3;SG;PRS	1	0	dis-diss
total		9	21	

Error Analysis

- Creative Pattern Errors:

Pattern	M.1	M.2	Example
V;V.PTCP;PRS			
double last letter + ng	8	1	slumppng
+nng	1	9	bownng
+iing	8	14	swimiing
V;V.PTCP;PST & V;PST			
double last letter + d	12	10	dieselld, drooppd
w+nd	2	1	vownd
+eed	5	7	renoveleed, unhateed
V;3;SG;PRS			
delete -c/d double letter before -c/d +s or +es	0	14	flood - flooos, rind - rinns, overpunch - overpunnhes
-th +e	3	0	outworthe
-ch +ss	0	2	bachss

Conclusion

- Comparing to previous models, Model 1 and Model 2 made more silly errors. Model 1 made less allomorphy errors.
- Model 1 and 2 also made creative pattern errors. Most of them associated with double spelling rule in English.
- Model 1 and 2 learned some existing linguistic patterns as well as creating new non-linguistic patterns.

Experiment 2. Research Questions

- How many input does the model need to learn a pattern?
  1. Pattern token sensitive: is sensitive to the absolute number of pattern tokens regardless of pattern ratio.
     e.g. A pattern can be learned based on 5 regular items out of total 10 items. It can also be learned based on 5 out of 15 or 20 total items.
  2. Pattern ratio sensitive: is sensitive to the ratio of patterns in total items.
     e.g. A pattern can be learned when patterns ratio is over 50%. If there are 10 items, 5 regular items are needed. If there are 100 items, 50 regular items are needed.
  3. Interaction of pattern token and ratio

Data

- Since Model 1 and 2 mostly failed to learn double spell rule, more double spell inflections were added to V;PST and V;V.PTCP;PRS types
- Added data are from CELEX database
- Enhanced 20: adding 20 items for each character-level pattern, e.g. 20 '-ked', 20 '-bed', 20 '-ling', 20 '-ming'...
- Enhanced 50: adding 50 items for each character-level pattern
- Regular pattern tokens remain the same, but ratio drops
- Double spell pattern tokens and ratio increase

Data Description

	V;PST	V;V.PTCP;PRS
Model 2		
Regular Token	817	949
Train Token	2016	1959
Regular %	40.53%	48.44%
double spell	134	166
double spell%	6.65%	8.47%
Enhanced 20		
Training	2182	2146
Regular %	37.44%	44.22%
double spell	301	353
double spell%	13.79%	16.45%
Enhanced 50		
Training	2346	2318
Regular %	34.83%	40.94%
double spell	465	525
double spell%	19.82%	22.65%

- Enhanced 20 and Enhanced 50 are trained on the same model as Model 2

Results: V;PST Accuracy

- double spell: ↑, regular: ↓, -d: –

Model 2

pattern	diff	✓	✗	Acc	train token	train ratio
regular	-ed	98	6	0.94	817	40.53%
-d	-d	86	3	0.97	856	42.46%
double spell		1	10	0.09	93	4.61%
	-bed	0	1	0.00	7	0.35%
	-ded	1	0	1.00	9	0.45%
	-ged	0	1	0.00	18	0.89%
	-ked	0	1	0.00	4	0.20%
	-led	0	5	0.00	20	0.99%
	-ned	0	1	0.00	12	0.60%
	-ted	0	1	0.00	23	1.14%

Enhance 20

pattern	diff	✓	✗	Acc	train token	train ratio
regular	-ed	72	32	**0.69**	817	37.44%
-d	-d	87	2	0.98	856	39.23%
double spell		4	7	0.36	77	3.53%
	-ged	1	0	1.00	38	1.74%
	-ked	1	0	1.00	7	0.32%
	-ned	1	0	1.00	32	1.47%

Enhance 50

pattern	diff	✓	✗	Acc	train token	train ratio
regular	-ed	43	61	**0.41**	817	34.83%
-d	-d	85	4	0.96	856	36.49%
double spell		3	8	0.27	157	6.69%
	-ded	0	1	0.00	9	0.38%
	-ged	1	0	1.00	68	2.90%
	-ked	1	0	1.00	7	0.30%
	-ted	1	0	1.00	73	3.11%

Results: V;V.PTCP;PRS Accuracy

- double spell: −, regular: ↓, e-ing: −

Model 2

rule	diff	✓	✗	Acc	train tokens	train ratios
regular	-ing	99	2	0.98	949	48.44%
e-ing	e-ing	84	0	1.00	831	42.42%
double spell		3	13	0.19	155	7.91%
	-bing	1	2	0.33	13	0.66%
	-ging	0	1	0.00	17	0.87%
	-ling	0	3	0.00	29	1.48%
	-ming	0	2	0.00	12	0.61%
	-ning	0	1	0.00	16	0.82%
	-ping	1	1	0.50	30	1.53%
	-ting	1	3	0.25	38	1.94%
special	e-ting	0	1	0.00	0	0.00%

Enhance 20

rule	diff	✓	✗	Acc	train tokens	train ratios
regular	-ing	52	49	0.51	949	44.22%
e-ing	e-ing	81	3	0.96	831	38.72%
double spell		3	13	0.19	175	8.15%
	-bing	0	3	0.00	33	1.54%
	-ling	1	2	0.33	29	1.35%
	-ming	1	1	0.50	12	0.56%

Enhance 50

rule	diff	✓	✗	Acc	train tokens	train ratios
regular	-ing	53	48	0.52	949	40.94%
e-ing	e-ing	79	5	0.94	831	35.85%
double spell		3	13	0.19	334	14.41%
	-bing	3	0	1.00	58	2.50%
	-ling	2	1	0.67	70	3.02%
	-ming	1	1	0.50	38	1.64%
	-ping	1	1	0.50	80	3.45%
	-ting	0	4	0.00	88	3.80%

Interim Results

- Evidence for Pattern Token sensitive:
  - With increased double spell patterns, double spell accuracy increases in V;PST
  - -d and -e+ing patterns remains about the same
- Evidence for Pattern Ratio sensitive:
  - Regular accuracy pattern tokens remained the same, but accuracy dropped significantly
  - Double spell accuracy remains the same for V;V.PTCP;PRS
- It's likely to be the interaction of pattern token and ratio

Error Analysis

	Enhanced 20	Enhanced 50	Model 2
V;V.PTCP;PRS			
double last letter + ng	49	48	1
+iing	1	4	14
triple last letter + ng	16	5	∅
V;PST			
double last letter + d	19	46	10
+eed	1	0	7
triple last letter + d	6	7	∅

1. +iing and +eed reduced, but double spell +ng/+d significantly increased
2. New pattern: Triple spell, e.g. whitennng, yodellld
3. It seems like the model has trouble processing vowels in patterns

Conclusion

- By increasing double spell patterns in training data, the double spell rule was not enhanced.
- The creative double spell last letter ‘+ng’/‘+d’ rule got significantly enhanced.
- Even created a new pattern: triple spell
- The double spell accuracy slightly increased or remained the same, but the regular accuracy dropped drastically.
- The ‘-d’ and ‘-e+ing’ pattern remained the same.
- These evidence suggest model might have trouble processing vowel in patterns.

Discussion

- What kind(s) of patterns did model learn?
  1. Some existing linguistic patterns
  2. The model creates robust new patterns. Increasing training pattern tokens enhanced the new patterns but not necessarily the targeted linguistic pattern.
- How many input does the model need to learn a pattern?
  1. The model is sensitive to both pattern token and pattern ratio.
  2. Increasing pattern tokens do enhance pattern learning, but not necessarily the target pattern.
- More questions:
  1. How did the model come up with triple spell rule?
  2. Why did the model have trouble processing vowel in patterns?

References

Calderone, B., Hathout, N., and Bonami, O. (2021). Not quite there yet: Combining analogical patterns and encoder-decoder networks for cognitively plausible inflection. *arXiv preprint arXiv:2108.03968*.

Corkery, M., Matusevych, Y., and Goldwater, S. (2019). Are we there yet? encoder-decoder neural networks as cognitive models of english past tense inflection. *arXiv preprint arXiv:1906.01280*.

Gorman, K., McCarthy, A. D., Cotterell, R., Vylomova, E., Silfverberg, M., Markowska, M., et al. (2019). Weird inflects but ok: Making sense of morphological generation errors. In *CoNLL 2019-23rd Conference on Computational Natural Language Learning*, *Proceedings of the Conference*. The Association for Computational Linguistics.

Kirov, C. and Cotterell, R. (2018). Recurrent neural networks in linguistic theory: Revisiting pinker and prince (1988) and the past tense debate. *Transactions of the Association for Computational Linguistics*, 6:651–665.

Wu, S., Cotterell, R., and Hulden, M. (2020). Applying the transformer to character-level transduction. *arXiv preprint arXiv:2005.10213*.