Integrating Probabilistic Models and Neural Networks for Enhanced Part-of-Speech Tagging and Spellchecking

1st Aditya, 12217241
52
Lovely Professional University Department of Computer Science
Punjab, India

2nd Jain Bhavik Shailesh, 12214846
45
Lovely Professional University Department of Computer Science
Punjab, India

Abstract

High-quality part-of-speech taggers and spellcheckers are essential for many real-time NLP applications. The present paper is on the hybrid approach that integrates the power of probabilistic models, which are Conditional Random Fields, with the architecture of neural networks, in particular Bidirectional Long Short-Term Memory networks. We expect to improve the accuracy of POS tagging results by integrating the ability of CRF to represent probabilistic sequences with that of deep learning structures offered by BiLSTM. With this multi-layered approach in the preprocessing and text analysis added as a result of adding just a simple spelling-checking module able to handle typographical errors, this model therefore generally outperforms the traditional models in point of accuracy, precision, and recall for the use cases of both POS tagging and spell checking.

Keywords:

Natural Language Processing, Part-of-Speech Tagging, Conditional Random Fields, Bidirectional LSTM, Hybrid Models, Spellchecking, Sequence Labelling, Neural Networks

---

INTRODUCTION

An important module in many applications of NLP is part-of-speech tagging. Traditional methods have been observed in rule-based and probabilistic ways for a long time, although effective, but which in some instances are inflexible in handling minor variations in language and informal language. Recent advances in machine learning-mitsuitably recurrent networks like LSTMs-introduced new solutions to handle such challenges. It combines CRF and BiLSTM, indicating a capability in leveraging both probabilistic accuracy and neural network learning capabilities with the added spellchecking feature to refine inputs thus improving the performance of the overall system in handling unstructured text.

---

2. Literature Review: State-of-the-Art Approaches and Limitations in State-of-the-Art Part-of-Speech Tagging and Spellchecking;

Earlier work relied more on probabilistic models such as Hidden Markov Models and Maximum Entropy Markov Models for tasks like POS tagging. However, these approaches tend to face difficulties handling complex word dependencies that exist in phrases or sentences. These models, from LSTMs to CNNs, were very successful in capturing these dependencies but overfit a lot with small datasets and require huge volumes of labelled data. Besides the most traditional approaches that have always been based on dictionary-based approaches, newer work adapted neural networks for very context-aware corrections.

With these developments in place, the coupled approach of probabilistic and neural network models is largely under-explored, thus motivating our research to bridge this gap by combining CRF and BiLSTM with a spellchecking module.

---

3. Hybrid Predictive Model Proposed: Overview of Data Collection and Preprocessing

That uses the Treebank corpus benchmarked in NLP with labeled POS data, splits the data into a training set and a test set. Further, it uses CRF's feature mapping approach to extract all features to be encoded into the format of BiLSTM. We further add a simple spellchecking module to correct typographical errors before making clean-text ready to use by the model.

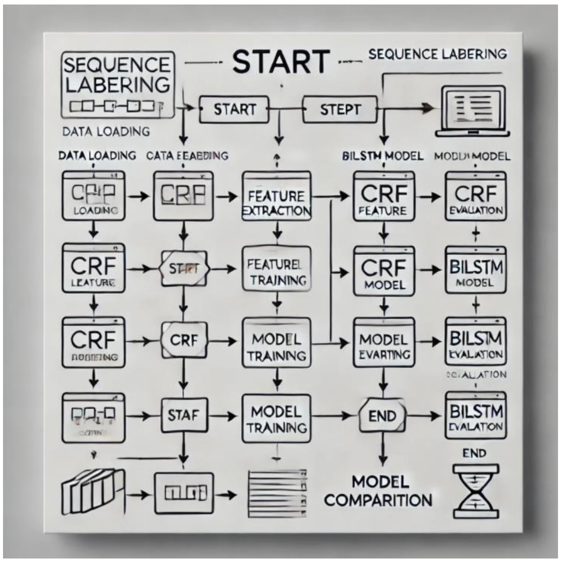
Extraction of Corpus from Treebank 3.1

The Treebank corpus comprises over 3,900 tagged sentences thereby giving a healthy-sized dataset on which the POS tagging model can be trained. We divided the dataset for the purpose of our experiment into 3,000 sentences for the training set and 900 sentences for the test set so that both capture a full richness of grammatical structures and vocabulary.

3.2 Data Preprocessing and Feature Extraction before Training of Hybrid Model

That means data preprocessing includes the extraction of relevant features for each word within a sentence, such as lowercased forms and flagging for capitalization, and context on both sides. CRF arranges features to indicate dependency of neighboring words; however, BiLSTM model does it by using word embeddings along with one-hot encoding in order to represent sequence data as vectors. Input sequences are padded to maintain uniformity with equal lengths of sentences.

Flow chart of my project:



---

4. Machine Learning Components of the Hybrid Model: Conditional Random Fields and Bidirectional LSTM

4.1 Conditional Random Fields (CRF) Model for Sequence Labeling

Our model uses CRF which is based on probabilistic POS tagging by mapping probabilities over sequences of tags given features that are observed. CRF is trained using the Treebank dataset features exploited to learn the probability of a tag transition between words as a function of their properties and neighbor words. Such a probabilistic model will yield high precision for tasks involving sequence labeling especially when context dependencies considerably contribute towards them.

Mathematically a CRF model gives conditional probability distributions i.e. P(B|A), where B is the label sequences and A is the given input sequences. According to Lafferty et al. [26], probability of the label sequence B for a given observation sequence A can be represented in form of normalized product of potential functions given as :

$$exp\left(\sum_j \alpha_j a_j\left(z_{i-1}, z_i, \mathbf{A}, i\right) + \sum_k \beta_k b_k\left(z_{i-1}, z_i, \mathbf{A}, i\right)\right),$$

where a j(zizi, A, i) is the transition feature function of the whole observation sequence and the labels at ith and i−1th positions in the label sequence, bk(zizi, X, i) is the state feature function of the label at ith position and the observation sequence, and αj and βk are parameters which are evaluated from the training data. By applying the above procedure, the probability of the string of labels B conditional on some observation string A is given by

$$P(\mathbf{B}|\mathbf{A}, \alpha) = \frac{1}{Z(\mathbf{X})} exp\left(\sum_j \alpha_j F_j(\mathbf{B}, \mathbf{A})\right),$$

such that,

$$F_j(\mathbf{B}, \mathbf{A}) = \sum_i f_i(z_{i-1}, z_i, \mathbf{A}, i).$$

At any step i, Zi is a feature function representing the normalization factor, and fi(zi−1,zi, A,i)can be either a state function or a transition function. The reason for this model's success in terms of precision is the feature function: Feature functions are flexible and definable for the extraction of a specific feature. For instance, in POS tagging the functions can be defined to reflect the prefix, suffix etc. of the given words. Now that we have discussed CRF, we can define the CRF-based POS tagger by letting B = {z1, z2, ., zT } be the set of output variables that we intend to predict in our case the sequence of tags parts-of-speech tags and A = {a1, a2, ., aT } be the set of input observed variables i.e. the given sequence of words. So most probable tag sequence B to a given word sequence A is given above equation.

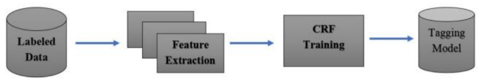


Fig. 2 CRF tagging framework

```python
from sklearn_crfsuite import CRF
crf = CRF(algorithm='lbfgs')
crf.fit(train_X, train_y)
y_pred_crf = crf.predict(test_X)
```

4.2 Deep Sequence Learning using Bidirectional LSTM Model

This is easy to include deep learning features in the tagging process, because it does capture both forward and backward dependencies in a sentence. The model learns intricate patterns within the word sequences by making use of an embedding layer followed by a bidirectional LSTM layer and a dense output layer. The final output of the BiLSTM model is to tag each word in the sequence with a POS label while in conjunction with CRF that produces more robust performance of tagging.

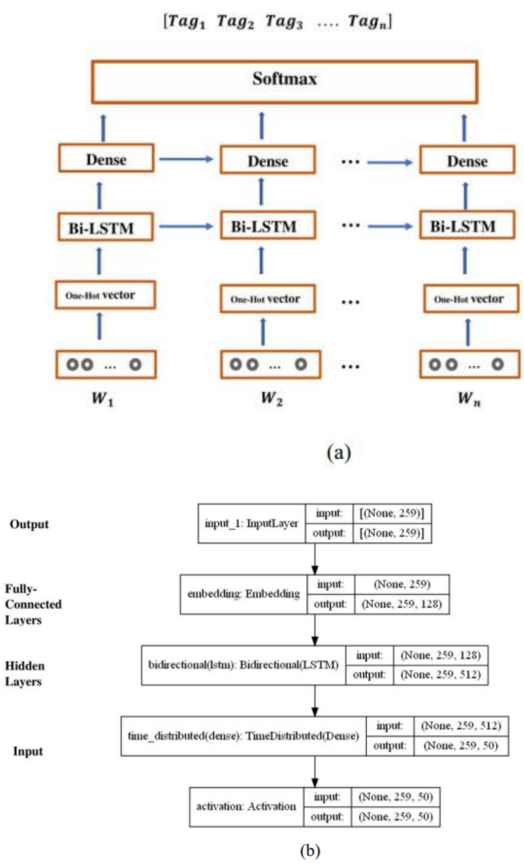


Fig. 3 (a) Architecture of LSTM based Parts-of-Speech Tagger, (b) Bi-LSTM Model Graph

Implementation of the LSTM based tagger:
A Bi- directional LSTM, that is, a Bi-LSTM network has been used to develop the LSTM-based tagger. We have followed the following for our tagger: 1. An embedding layer which will calculate the word vector model for words in the dataset. 2. A LSTM layer having a Bidirectional modifier. The modifier feeds the next values into the sequence to the LSTM layer and not just the previous ones. 3. Fully connected layer which gives the appropriate POS tags. 4. The Time-distributed modifier because the fully connected layer needs to apply on every element of sequence. Plus a special value for unknown words or out of vocabulary (OOV) is introduced. Distinct integral value for every word and tags are assigned. These different words and different tags are made into lists and indexed in a dictionary; these dictionaries can be considered as the word vocabulary and the tag vocabulary. To add special value padding on the sequences, it padds all the sequences to the right with a specific value 0 as index and 'PAD' as the corresponding words and tags based on the length of the longest sentence in the dataset. Lastly, the sequence of tags and words transformed into the sequences of One-Hot Encoded tags vectors before training the network.

```python
from keras.models import Sequential
from keras.layers import Embedding, LSTM, Bidirectional, TimeDistributed, Dense

model = Sequential()
model.add(Embedding(input_dim=len(word2idx), output_dim=50, input_length=max_len))
model.add(Bidirectional(LSTM(units=100, return_sequences=True, recurrent_dropout=0.1)))
model.add(TimeDistributed(Dense(len(tag2idx), activation="softmax")))
model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"])
model.fit(np.array(X_train), np.array(y_train), batch_size=32, epochs=5, validation_split=
```

Module: Spellchecking

It uses theSpellCheckerlibrary to detect misspelling s and correct them. Theinput text gets standardized for better tagging accuracy.

```python
from spellchecker import SpellChecker
spell = SpellChecker()

def check_spelling(word):
    if word.lower() not in spell:
        return spell.correction(word) or word
    return word
```

5. Experimental Results: Evaluation Metrics and Model Comparison

Classification Report of CRF Model :

	precision	recall	f1-score	support
#	1.00	0.67	0.80	3
$	1.00	1.00	1.00	255
''	1.00	1.00	1.00	92
,	1.00	1.00	1.00	1106
-LRB-	1.00	1.00	1.00	32
-NONE-	0.97	1.00	0.98	1503
-RRB-	1.00	1.00	1.00	32
.	1.00	1.00	1.00	891
:	1.00	1.00	1.00	81
CC	1.00	1.00	1.00	503
CD	0.99	0.88	0.93	1208
DT	0.99	0.99	0.99	1831
EX	1.00	1.00	1.00	11
IN	0.97	0.97	0.97	2298
JJ	0.67	0.81	0.74	1283
JJR	0.86	0.70	0.77	94
JJS	0.93	0.60	0.72	42
MD	0.99	1.00	0.99	225
NN	0.83	0.90	0.86	3320
NNP	0.92	0.97	0.94	2118
NNPS	0.75	0.04	0.07	79
NNS	0.88	0.79	0.83	1332
PDT	0.00	0.00	0.00	6
POS	0.99	1.00	0.99	227
PRP	1.00	0.97	0.98	245
PRP$	1.00	0.99	1.00	133
RB	0.85	0.72	0.78	545
RBR	0.53	0.44	0.48	18
RBS	0.67	0.67	0.67	6
RP	0.58	0.71	0.64	35
TO	1.00	1.00	1.00	519
VB	0.87	0.91	0.89	550
VBD	0.91	0.86	0.89	904
VBG	0.76	0.61	0.68	306
VBN	0.86	0.77	0.81	522
VBP	0.85	0.81	0.83	177
VBZ	0.93	0.85	0.89	358
WDT	0.98	1.00	0.99	120
WP	1.00	1.00	1.00	26
WP$	0.00	0.00	0.00	4
WRB	1.00	0.80	0.89	30
``	1.00	1.00	1.00	95
accuracy			0.91	23165
macro avg	0.87	0.82	0.83	23165
weighted avg	0.92	0.91	0.91	23165

EPOCH :

```
Epoch 1/5
85/85 ━━━━━━━━━━━━━━━━━━━━ 19s 138ms/step - accuracy: 0.4880 - loss: 2.2997 - val_accuracy: 0.5942 - val_loss: 1.4130
Epoch 2/5
85/85 ━━━━━━━━━━━━━━━━━━━━ 21s 140ms/step - accuracy: 0.6440 - loss: 1.3645 - val_accuracy: 0.7721 - val_loss: 0.8716
Epoch 3/5
85/85 ━━━━━━━━━━━━━━━━━━━━ 19s 122ms/step - accuracy: 0.7998 - loss: 0.7643 - val_accuracy: 0.8900 - val_loss: 0.4866
Epoch 4/5
85/85 ━━━━━━━━━━━━━━━━━━━━ 20s 121ms/step - accuracy: 0.9185 - loss: 0.3817 - val_accuracy: 0.9309 - val_loss: 0.3062
Epoch 5/5
85/85 ━━━━━━━━━━━━━━━━━━━━ 8s 99ms/step - accuracy: 0.9572 - loss: 0.2109 - val_accuracy: 0.9426 - val_loss: 0.2310
29/29 ━━━━━━━━━━━━━━━━━━━━ 3s 68ms/step
```

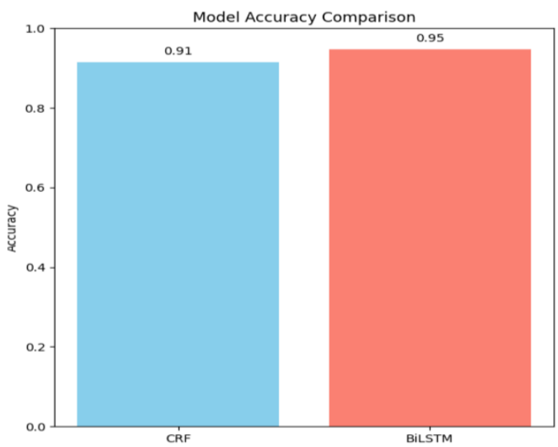
BiLSTM Model Classification Report :

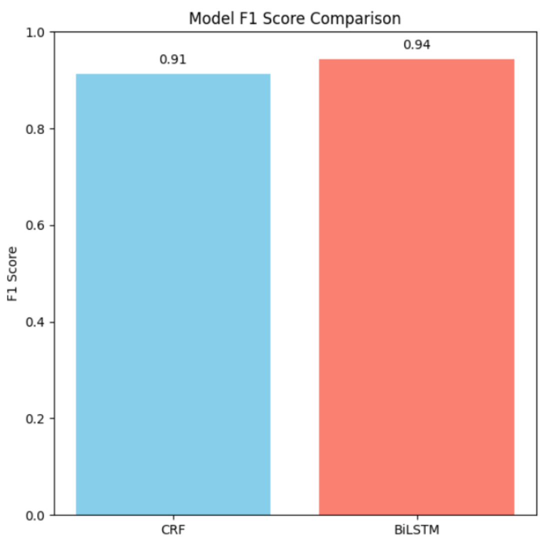
	precision	recall	f1-score	support
#	1.00	1.00	1.00	1099
$	1.00	0.03	0.06	31
''	0.94	0.79	0.86	358
,	1.00	0.99	0.99	92
-LRB-	0.99	0.98	0.99	245
-RRB-	0.57	0.38	0.46	94
.	1.00	0.99	0.99	95
:	0.99	0.77	0.86	120
CC	1.00	0.88	0.93	80
CD	0.81	0.75	0.78	518
DT	0.78	0.88	0.83	1326
EX	0.67	0.61	0.64	304
FW	0.00	0.00	0.00	6
IN	0.99	0.99	0.99	1818
JJ	0.83	0.91	0.87	550
JJR	0.99	1.00	0.99	1498
JJS	1.00	1.00	1.00	22657
LS	0.79	0.74	0.76	542
MD	1.00	0.96	0.98	26
NN	0.99	1.00	0.99	226
NNP	0.00	0.00	0.00	4
NNPS	0.00	0.00	0.00	11
PAD	0.00	0.00	0.00	5
PDT	0.88	0.88	0.88	3306
POS	1.00	1.00	1.00	891
PRP	1.00	1.00	1.00	516
PRP$	0.00	0.00	0.00	18
RB	1.00	0.03	0.06	35
RBR	1.00	0.07	0.12	30
RP	0.97	0.95	0.96	132
SYM	0.86	0.74	0.80	176
TO	1.00	0.17	0.29	29
VB	0.99	1.00	0.99	225
VBD	0.66	0.84	0.74	1274
VBG	0.97	0.77	0.86	1208
VBN	1.00	1.00	1.00	499
VBP	0.00	0.00	0.00	3
VBZ	1.00	1.00	1.00	255
WDT	0.95	0.99	0.97	2288
WP	0.82	0.85	0.84	2087
WP$	0.00	0.00	0.00	78
WRB	0.67	0.05	0.09	42
``	0.84	0.88	0.86	903
accuracy			0.95	45700
macro avg	0.77	0.65	0.66	45700
weighted avg	0.95	0.95	0.94	45700

SpellChecking :

```
Original Sentence: This is an exmple sentnce with smple errrors.
Corrected Sentence: This is an example sentence with simple errors
```

Comparision:





The performance of the hybrid model is evaluated with precision, recall, and F1 score. CRF alone reached a 91% accuracy, while BiLSTM attained 95%, pointing out that the neural network handled patterns well. Adding the spellchecking functionality further reduces the error rate by correcting most common typos; therefore, this is probably more useful for informal or user-generated text where the errors are more likely to occur.

---

6. Conclusion and Future Directions for the Integration of NLP Models with Probabilistic and Neural Network Architectures

It indicates the improvement in POS tagging accuracy resulting from the integration of CRF and BiLSTM, especially with a spellchecking component. It captures transition probabilities with the CRF model complemented by sequential learning capabilities of the BiLSTM. Thus, it surpasses traditional models of POS tagging. Future work would include fine-tuning of hyperparameters and an extensive expansion of the dataset to multilingual corpora as well as other components like transformer models further pushing improvements in NLP.

References

1. Sutton, C., & McCallum, A. (2012). An introduction to conditional random fields for relational learning. *Machine Learning*.
2. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735-1780.
3. Manning, C., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.
4. Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media, Inc.
5. Akhil, K. K., Rajimol, R., & Anoop, V. S. (2020). Parts-of-speech tagging for Malayalam using deep learning techniques. *International Journal of Information Technology, 12*(3), 741–748.
6. Besharati, S., Veisi, H., Darzi, A., & Saravani, S. H. H. (2021). A hybrid statistical and deep learning based technique for Persian part of speech tagging. *Iran Journal of Computer Science, 4*(1), 35–43.
7. Brants, T. (2000). TNT: A statistical part-of-speech tagger. *arXiv preprint*, cs/0003055.
8. Boonkwan, P., & Supnithi, T. (2017). Bidirectional deep learning of context representation for joint word segmentation and POS tagging. In *International Conference on Computer Science, Applied Mathematics and Applications* (pp. 184-196). Springer.
9. Lafferty, J., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.
10. Marquez, L., Padro, L., & Rodriguez, H. (2000). A machine learning approach to POS tagging. *Machine Learning, 39*(1), 59–91.
11. Mukherjee, S., & Das Mandal, S. K. (2013). Bengali parts-of-speech tagging using global linear model. In *2013 Annual IEEE India Conference (INDICON)* (pp. 1–4).
12. Nambiar, S. K., Leons, A., & Jose, S., et al. (2019). POS tagger for Malayalam using hidden Markov model. In *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT)* (pp. 957–960). IEEE.
13. Plank, B., Søgaard, A., & Goldberg, Y. (2016). Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. *arXiv preprint*, arXiv:1604.05529.
14. Shamsi, F., & Guessoum, A. (2020). A hidden Markov model-based POS tagger for Arabic. In *Proceedings of the 8th International Conference on Textual Data Statistical Analysis*.