Common data sources

ACL

EMNLP sas

ICCASR

Interspeech

<https://www.opensubtitles.org/en/search>

Switchboard corpus

IMDA National Speech Corpus

Librespeech etc. not very useful since its just reading of books or articles, might as well just use open wiki etc. with more data.

En-cr TEDtalks or [TED – Ultimate Dataset | Kaggle](https://www.kaggle.com/miguelcorraljr/ted-ultimate-dataset) or Open speech more relavant.

Progress for 9-Dec-20: Cleanup of TED Ultimate Dataset- Removal of speaker name before semicolons, removal of round brackets, TODO: removal of square brackets (split using square brackets, compare case before after).

10-Dec-20: Completed TED cleanup.

Punctuation to predict: No Apostrophes. Include all

Lvl 1: .,?!- – —;: // Include all possibly readable symbols as text

Lvl 2: round brackets, square brackets, quotation marks

Leading Punctuation

Don’t handle caps together, things like Acronyms cannot be processed in the same manner?

~~TEDLIUM3 (contains audio to text without punctuation) example of output of many ASR systems.~~

~~Key features of TEDLIUM3 (Jun 2019):~~

~~207 hours -> 452 hours of audio (not relavant atm).~~

~~2.2M -> 4.9M words | 92976 -> 268231 word segments | 1495 -> 2351 talks~~

~~Autoalignment of audio files (.sph) with transcripts (.stm) using Kaldi.~~

Rich Transcription Categories

Speaker diarization

Sentence segmentation

Punctuation recovery or detection

Capitalization / truecasing

Disfluency detection and filtering:

1. [Recovering Capitalization and Punctuation Marks on Speech Transcriptions (inesc-id.pt)](https://www.inesc-id.pt/publications/4467/pdf) Recovering Capitalization and Punctuation Marks on Speech Transcriptions
2. [2004.00248.pdf (arxiv.org)](https://arxiv.org/pdf/2004.00248.pdf#:~:text=A%20pre-trained%20bidirectional%20encoder%20represen-%20tations%20from%20transformers,to%20learn%20task%20invariant%20knowledge%20for%20punctuation%20prediction.) Adversarial Transfer Learning for Punctuation Restoration 2020
3. [Robust Prediction of Punctuation and Truecasing for Medical ASR (aclweb.org)](https://www.aclweb.org/anthology/2020.nlpmc-1.8.pdf) 2020
4. [The Use of Prosody in a Combined System for Punctuation Generation and Speech Recognition (isca-speech.org)](https://www.isca-speech.org/archive/archive_papers/eurospeech_2001/e01_2757.pdf) Kim and Woodland 2001
5. [IEEE Xplore Full-Text PDF:](https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6423471) PROSODY-BASED SENTENCE BOUNDARY DETECTION IN CHINESE BROADCAST NEWS 2012
6. [IEEE Xplore Full-Text PDF:](https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9054337) IMPROVING PROSODY WITH LINGUISTIC AND BERT DERIVED FEATURES IN MULTI-SPEAKER BASED MANDARIN CHINESE NEURAL TTS 2020
7. [GitHub - nkrnrnk/BertPunc: SOTA punctation restoration (for e.g. automatic speech recognition) deep learning model based on BERT pre-trained model](https://github.com/nkrnrnk/BertPunc)
8. [Bidirectional Recurrent Neural Network with Attention Mechanism for Punctuation Restoration (semanticscholar.org)](https://pdfs.semanticscholar.org/8785/efdad2abc384d38e76a84fb96d19bbe788c1.pdf) 2016 INTERSPEECH
9. [download (psu.edu)](https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.28.276&rep=rep1&type=pdf) PERFORMANCE MEASURES FOR INFORMATION EXTRACTION
10. [RESTORING PUNCTUATION AND CAPITALIZATION IN TRANSCRIBED SPEECH (googleusercontent.com)](https://static.googleusercontent.com/media/research.google.com/en/pubs/archive/34562.pdf) 2009
11. [1908.02404.pdf (arxiv.org)](https://arxiv.org/pdf/1908.02404.pdf) Fast and Accurate Capitalization and Punctuation for Automatic Speech Recognition Using Transformer and Chunk Merging 2019
    1. Overlap chunk split, Transform, Overlap chunk merge (0.5 overlap)
    2. Utilised on on
12. [CMSY9 (iwslt.org)](https://workshop2015.iwslt.org/downloads/IWSLT_2015_RP_16.pdf) Punctuation Insertion for Real-time Spoken Language Translation 2015
13. Fastpunct? (no as lacking model or paper) Or bertpunc
14. [Leveraging a Character, Word and Prosody Triplet for an ASR Error Robust and Agglutination Friendly Punctuation Approach (isca-speech.org)](https://www.isca-speech.org/archive/Interspeech_2019/pdfs/2132.pdf) 2019 interspeech
15. [Evaluating an automatic speech recognition service | AWS Machine Learning Blog (amazon.com)](https://aws.amazon.com/blogs/machine-learning/evaluating-an-automatic-speech-recognition-service/) For end 2 end asr, talks about evaluation.
16. [1805.04699.pdf (arxiv.org)](https://arxiv.org/pdf/1805.04699.pdf) TEDLIUM3
17. [2012.02012.pdf (arxiv.org)](https://arxiv.org/ftp/arxiv/papers/2012/2012.02012.pdf) end 2 end ASR
18. [5.3: Proofreading for Punctuation – Communication at Work (pressbooks.pub)](https://ecampusontario.pressbooks.pub/communicationatwork/chapter/5-3-proofreading-for-punctuation/)
19. [Visualizing Models, Data, and Training with TensorBoard — PyTorch Tutorials 1.7.1 documentation](https://pytorch.org/tutorials/intermediate/tensorboard_tutorial.html)

<https://arxiv.org/pdf/1905.06791.pdf> unsupervised asr and tts

Implement HMM

Does pretrained wiki bert perform on speech tasks?

Importance of punctuation in bert

Different types of punctuation, various uses of comma etc. (1 pg. 40)

Sentence punctuation (, . ! ? ellipsis) used as part of phrasing (generally succeeded by a space or quote)

Quotation marks or brackets

Sentence punctuation before quotation marks

To train sentence punct classifier:

Strip all non-sentence or word punctuation (brackets, quotation marks),

Tokenize,

assign word level sentence punctuation classes,

try to implement a cnn classifier over glove embedding of the sentence

Else, train a seq2seq

TODO:

Bert finetune punctuation classifier

[1910.13461.pdf (arxiv.org)](https://arxiv.org/pdf/1910.13461.pdf) train encoder?

Perhaps modify bert inputs, cls for valid punctuation, mask punctuations

Randomly replace words with homonyms and recover the original word (if using encoder)

Randomly replace words with homonyms or synonyms to expand dataset (Is this necessary?)

Bert understands context of punctuation, it might be very useful?

Mask and return probability of all punctuation tokens, return token if above threshold.

Dealing with quotation marks: Perhaps simply ignore or insert in a 2nd pass.

Does differentiating open and close quote matter?

Check how bert input is represented (If I’m not wrong, just tokens and labels)

Which bert model? Try to find the smallest that performs.

Should there be a nospace class? No. no training data for it.

Create punct importance function

Randomly mask punctuation to be classified? Or start from 0 or start from max? How to do this efficiently?

Preprocess tedlium with some pipeline: merge lines, split into xx length sequences, (Determine xx), if sequence contains repeated non-apos punct, pop. Is it true that a smaller length

Try to run some trained bert punct and create evaluation metric and evaluate using the nsc or tedlium

Bert final model: Input: string with apostrophes, tokenize, classify by all punct tokens, loop through model until all classes are zeros.

Drawbacks: Will need to run each classifier +1 time. Perhaps using the chunk overlap/merging can help to double check and build the subsequent punctuations? What about a 2/3 overlap – 1st 1/3 to predict base punct, 2nd 1/3 to double check and last 1/3 to add.

Training (Input: String strip punct, layer with most impt punct, or keep most impt punct and classify 2nd most impt punct (or if really common, keep 1st 2 punct and classify 3rd punct?!)

Pros: the position of all punctuation marks relative to the entire sub-word space is already defined during the bert pre-training process.

Considerations:

-Do I have to normalize the pre-trained embedding to fix the context of spoken words? Perhaps look into Robust Prediction of Punctuation and Truecasing for Medical ASR to see how they perform transfer learning?

Their Approach:

-Perform some Masked Language Model Finetuning on domain specific data.

-Domain+Task Adaptation: 50% of mask being punctuation – not very useful since punctuation is so little, and it might reduce the amount of fit of the data to the new domain??

Truecase and punct predict simultaneously?

Data format:

Punctuation mask (assign classes to words with ending with none, full-stop, comma, close quote, openquote, semicolon, colon, qmark, xmark.

Issue: double punctuation i.e. Titles with punct, abbreviations op. cit.?

Periods (***..***), question marks (***??***), and exclamation marks (***!!***) do not double up. Why? Because that would look weird.

—A period never follows an exclamation mark or question mark.

“Don’t be absurd!” said Henry. . . . “You remember what the Hatter said to her: ‘Not the same thing a bit! Why you might just as well say that “I see what I eat” is the same thing as “I eat what I see”!’ ”

Evaluation

Precision, Recall, SER ( S + D + I + H?) / N or normalised Levenshtein edit distance , F Score, AUC

Ted datasource analysis:

Contains many speaker tags and

# Research papers

## [Bidirectional Recurrent Neural Network with Attention Mechanism for Punctuation Restoration (isca-speech.org)](https://www.isca-speech.org/archive/Interspeech_2016/pdfs/1517.PDF)

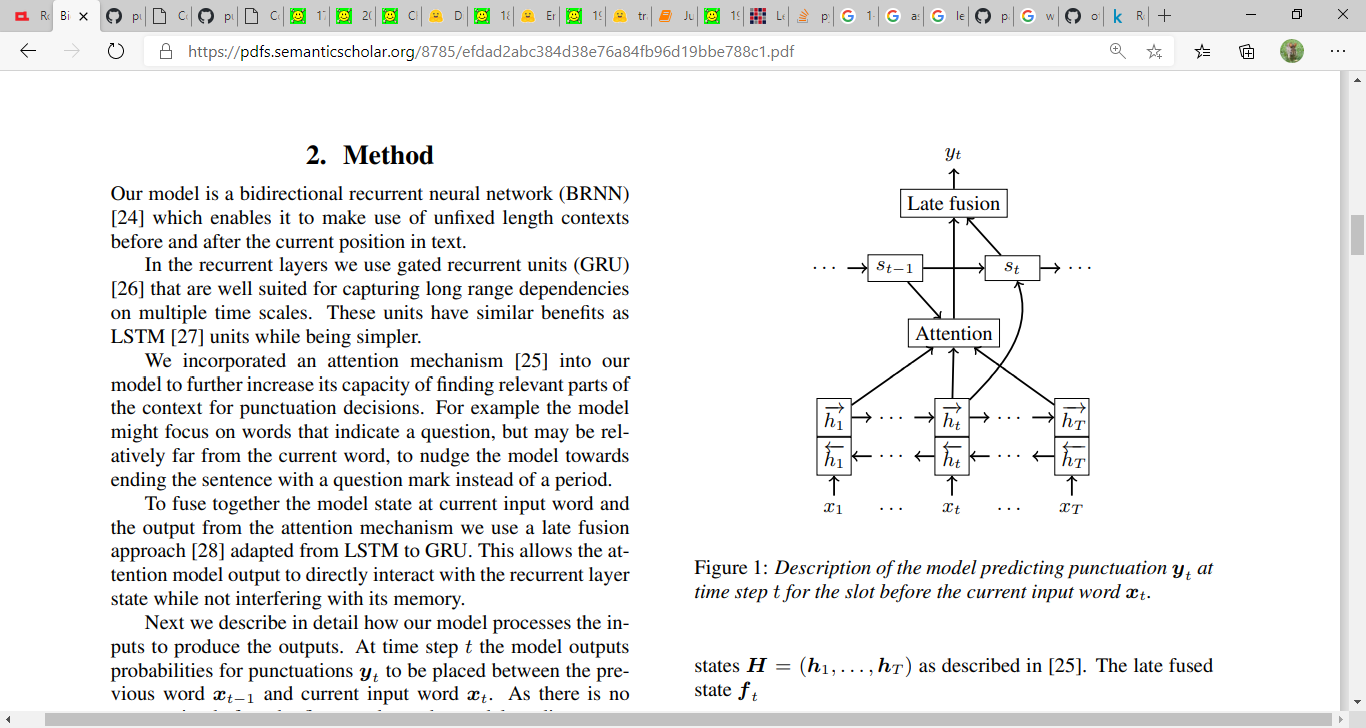
Code: [ottokart/punctuator2](https://github.com/ottokart/punctuator2)

8 Sep 2016

### Ideas

* Use of Attention Mechanism within Bidirectional RNN increases capacity to take context into consideration.
* Use GRUs within recurrent layers.

Model: Glove Embeddings-> Bidirectional GRU -> Attention with GRU -> Late fused attention and state -> softmax



For 2 stage (with pause duration), the fusion state output is fed into a GRU layer which produces the outputs.

### Training details

, . ? None

Optimiser: AdaGrad(lr=0.02), Gradient clipped to 2

Loss: Negative log-likelihood

Beam-search found to accumulate mistakes, ineffective.

Stage-1: Early-stopping with patience 0, stage-2: patience 5

Input length: 200, starts with beginning of sentence. Output sequence: 2nd word onwards.

Shuffled every epoch batch-size 128

Dataset: Estonian (334M word – newspapers and WWW, 1M word pause annotated transcripts.)

English (IWSLT dataset 2012 train 2.1M dev 296K words, IWSLT 2011 reference and ASR 13M word for test.)

Metrics: Precision, Recall, F1-score

### Result

The proposed model demonstrated improvement in all metrics for all classes compared to the previous DNN and CNN models.

Bidirectionality found to be most important factor in model. Attention is less critical and only affected the question mark class.

## [Joint Learning of Correlated Sequence Labelling Tasks Using Bidirectional Recurrent Neural Networks (arxiv.org)](https://arxiv.org/pdf/1703.04650.pdf)

18 Jul 2017

### Ideas

* Treating punctuation and capitalization as correlated tasks (Both tasks can benefit from each other’s output)
* An RNN (BiRNN)-based joint learning framework for multiple correlated sequence labeling tasks, with no feature engineering.
* Improvement in both punctuation and capitalisation prediction on speech transcripts by jointly training both punctuation and capitalization, without using any prosodic features.
* Joint training loss function (Weighted average of losses across all tasks)

### Datasets

* 60:30:10 train-validation-test split
* Intelligence squared debate transcripts (45 debates featuring 4 speakers each)
* IWSLT TED Talks (English transcript from English to French MT task)

### Model comparison:

Single-BiRNN vs Corr-BiRNN

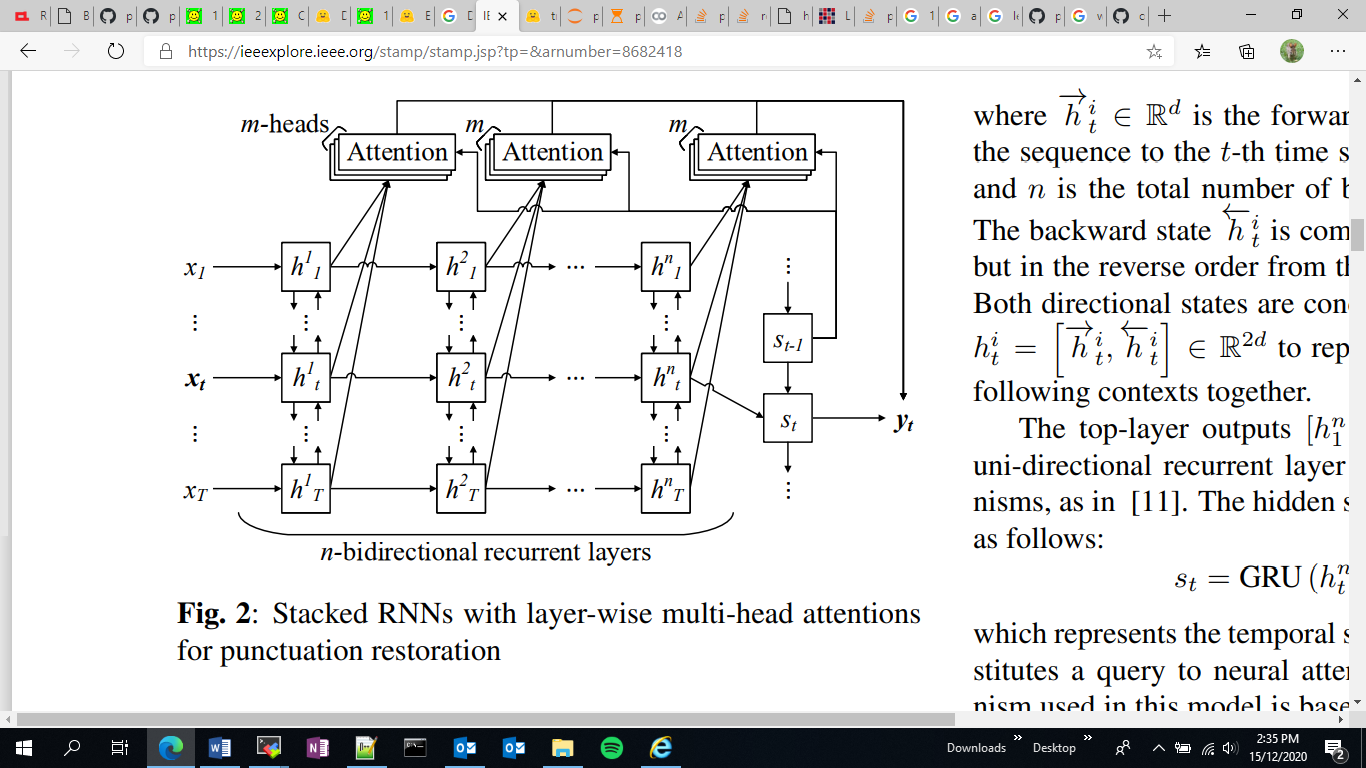
Punctuation Labels: (Period, Comma, ? mark, None); Caps labels: (Upper (merged with MixedCase), lower, CAPS)

### Data Pre-processing:

* Training example: 40-70 tokens, each example begins with new sentence, sentence need not be complete.

Evaluation: Levenshtein alignment of ASR output to transcript

## [Deep Recurrent Neural Networks with Layer-wise Multi-head Attentions for punctuation restoration](https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8682418)

Apr 2019

-Utilises multiple recurrent layers to learn more hierarchical contexts

-Multi-head attention applied on each layer

-Showed improvement in comparison to the baseline model T-BRNN: Bidirectional Recurrent Neural Network with Attention Mechanism for Punctuation Restoration (isca-speech.org)

Dataset: IWSLT

## [Fast and Accurate Capitalization and Punctuation for Automatic Speech Recognition Using Transformer and Chunk Merging (arxiv.org)](https://arxiv.org/pdf/1908.02404.pdf)

7 Aug 2019

### Overview: Boundary words may be less accurate than words in the centre of sequence, so cut off the ends of classification output and join the middle portions together.

### Ideas featured

* For inference, split input into overlapping chunks, process, then merge chunks to obtain output. This reduces the error rate near the boundaries of each chunk.
* Chunk merging demonstrated

Dataset: British National Corpus with 100M spoken and written words.

Pre-processing steps

* Strip all characters not in [A-z comma period question]
* Join punctuation to the preceding word.
* Split text into chunks with ½ stride.

Merge by cutting tail of first sequence and head of 2nd sequence and joining the 2 sequences

Comment: Other methods of merging can be explored, perhaps do a 1/3 stride and perform voting or merge using some other algorithm? This doubles the inference time, perhaps can merge this with the double punctuation prediction to make the process more efficient?

## [Adversarial Transfer Learning for Punctuation Restoration (arxiv.org)](https://arxiv.org/pdf/2004.00248.pdf)

1 Apr 2020

, . ?

Model: Pretrained-BERT 🡪 BLSTM Layer 🡪 CRF Layer

Uses Multi-task learning, with POS Tagging as an auxiliary task to improve performance on punctuation prediction

Adversarial discriminator used to prevent learning of task specific information in shared layers, allow for generalization across tasks.

Minimize negative log likelihood across all tasks.

Datasets used: IWSLT for punctuation, Penn Treebank for POS tagging task.

Evaluated with Precision, Recall, F1 Score.

Optimizer: For both BERT-CRF and BERT-BLSTM-CRF Adam with gradient clipping and warmup. The warmup steps = 4,000. The batch size = 32, dropout rate = 0.1. The initial learning rate = 5e-4.

Result: Addition of POS Tagging task improved on the model’s ability to classify punctuations.

## [Transfer Learning for Punctuation Prediction](https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9023200&casa_token=Nf3LHca0UtoAAAAA:uc02yurEQ7JaYcQgJ3HOJDIuT7kKKXrKWaO_8OmqYFc_gJmAj-MzlCg5Sl4E-Py_SFYnii0&tag=1)

Code: [panda-baba/bert\_punct: Punctuation restoration in ASR text (github.com)](https://github.com/panda-baba/bert_punct)

Treats punctuation prediction as sequence labelling task. Uses a 2-layer model. First layer: pre-trained BERT, 2nd layer: BLSTM with linear-chain CRF classifier.

Uses IWSLT 2012 (TED talks) dataset

Tokens: [ comma period question ]

Optimizer: Adam, lr 2e-5 beta\_1 0.9 beta\_2 0.999, weight\_decay 0.01, warmup 0.1%, exponential decay rate 0.9

Gradient clipping with threshold = 2, 5 epoch, batch size 8,

LSTM 200 dim, 6 output classes, sequence length 128, dropout 0.1.

Result: Model performs better than SAPR (model treating task as Machine Translation) overall, performs better on less common punctuation classes.

## [Adaptive Name Entity Recognition under Highly Unbalanced Data (arxiv.org)](https://arxiv.org/pdf/2003.10296.pdf)

10 Mar 2020

### Overview: Attempt to improve the classification of weaker classes in NER by training a separate weak classifier.

Models implemented: CRF, BERT, BLSTM-CRF

Proposed framework: Divide classes into Strong and Weak (Fewer data) class, identify weak classes using a RNN-CNN network with weighted cross entropy loss, train a weak and strong classifier, and combine the outputs of the three classifiers using majority voting.

Result: Method produced some improvement over the baseline versions (for BLSTM and BERT respectively) for weak classes.

Comments: This approach helps to deal with inadequacies in the training process due to highly imbalanced data, but the Dice Loss for Data-imbalanced NLP Tasks (arxiv.org) approach seems better with a less complicated architecture and might deal with the issue of imbalance as well?

## [Efficient Automatic Punctuation Restoration Using Bidirectional Transformers with Robust Inference (aclweb.org)](https://www.aclweb.org/anthology/2020.iwslt-1.33.pdf)

May 2020

, . ?

RoBERTa pretrained model followed by 2 linear models.

### Training Process

Trained with sliding window of 1 token.

Optimizer: LookAhead with RAdam, Cross-entropy loss

Best model: Prediction window size of 100, final layer dropout 0.2, 1500 hidden units.

Train the head for 9 epochs with batch-size 1000, before unfreezing the roberta layer and finetuning for 3 epoch, batch-size 250. (Managed to train a decent model with relatively high F1 score within 1 epoch of head and full respectively)

Lookahead: sync rate 0.5, sync period 6. RAdam: lr 1e-5, β1 = 0.9, β2 = 0.999, epsilon = 1e−8

IWSLT 2012 dataset

### Result

Lookahead with RAdam performed best compared to Adam/RAdam/LookAhead with Adam

Cross-entropy loss performed best compared to Focal loss / Class weighting.

Sequential Prediction requires ~15x longer inference times, with only a slight 2.2% performance benefit compared to parallel prediction without aggregation.

Parallel prediction with aggregation performs better but has a higher inference time.

DistilRoBERTa found to perform significantly worse than Roberta Large. I suspect that distilling the finetuned Roberta model would perform better than finetuning the DistilRoBERTa model.

## [Punctuation Restoration using Transformer Models for Resource-Rich and -Poor Languages (noisy-text.github.io)](http://noisy-text.github.io/2020/pdf/2020.d200-1.18.pdf)

Aug? 2020

### Overview: Demonstrate effectiveness of data augmentation in improving punctuation retrieval using BERT variants on English and Bangla Texts.

Looks into data augmentation strategy to deal with noise. Simulates three common forms of noise in ASR transcripts:

* Insertion: Add unknown token randomly
* Substitution: Replace token with unknown randomly
* Deletion: Delete token randomly

Finetune the probability of performing each of the 3 noise injection approaches.

Maximum sequence length of 256, with start and end token. Padded with padding token

Uses a batch size of 8 and shuffle the sequences before each epoch. Learning rates are 5e-6 for large models, and 1e-5 for base models

Datasets: IWSLT 2012 for English, Bangla newspapers and using G Cloud Speech API to transcribe Bangla short story audio data.

Models tested: BERT, RoBERTa, ALBERT, DistilBERT, mBERT, XLM-RoBERTa

Result:

* (Large | Monolingual) models faired better than base models
* RoBERTa faired best, optimal parameters: αIns = 0.15, αsub = 0.4, αdel = 0.4
* Experimented with using CRF after the linear layer for predicting the most probable tag sequence instead of using the softmax layer, but did not notice any performance improvement and even a slight decrease in ASR test data performance.

Evaluation: Possibly replace tokens with homophones to better simulate errors for substitution.

The point about CRF being ineffective can be re-evaluated.

## [Robust Prediction of Punctuation and Truecasing for Medical ASR (arxiv.org)](https://arxiv.org/pdf/2007.02025.pdf)

11 Jul 2020

### Overview: Evaluates effectiveness of several pretrained BERT variants in being transferred to the medical domain.

,.?

This paper focuses on the domain of Medical data.

Ideas featured:

* Effectiveness of pretrained BERT model vs model trained solely on medical data.
* Pretrained BERT consistently outperformed baseline BLSTM
* BioBERT / RoBERTa outperformed BERT base
* Subword models fared better than full-word models
* Importance of domain-relevant data

Model comparison:

* 2 Recurrent: 3 LSTM, 3 BLSTM (Baseline)
* 2 Non-recurrent: Character level CNN Highway [Character-Aware Neural Language Models (arxiv.org)](https://arxiv.org/pdf/1508.06615.pdf), Transformer, Encoder (Attention is all you need)
* Compared various BERT models- BERT, RoBERTa, BioBERT

Joint modelling of Case and Punctuation, just using lexical information

1. Text -- Wordpiece Tokenizer --> Subword embeddings
2. Subword embeddings – pretrained BERT encoder --> BERT embedding
3. BERT encoder output – Softmax (Wk hi + bk )--> punctuation labels
4. (BERT encoder output, punctuation labels) – Softmax (Wl (pi ⊕ hi) + bl ) --> case labels

Punctuation labels: (Period, Comma, ? mark, None); Caps labels: (Upper, lower, CAPS, MixedCase)

Dataset: Wiki dataset and internal Medical dataset

Objective: Maximise joint probability P (p1: T, c1: T|x1: T).

Loss: weighted average of cross entropy loss for P. and C. (L = α \* L p + L c). Optimal α was found to be 0.6

Training process:

Domain adaptation:

Finetune pretrained BERT on medical data, masking 15% of tokens

Task adaptation:

Finetune model on medical data, with 50% of the masked tokens being punctuations.

Data Augmentation:

* Transcribe provided audio with some available algorithm to increase word error up to a maximum of 25% WER., training on the data with highest WER and test and dev set being the lowest and next lowest 50.
* Overlapping and Chunking – split into chunks of 200 words and overlap of 50 words. [1908.02404.pdf (arxiv.org)](https://arxiv.org/pdf/1908.02404.pdf)

## [Multimodal Semi-supervised Learning Framework for Punctuation Prediction (arxiv.org)](https://arxiv.org/pdf/2008.00702.pdf)

3 Aug 2020

### Overview: Demonstrates that pretraining both lexical and acoustic encoders on unlabelled data can improve performance of punctuation retrieval.

-Utilises pre-trained lexical and acoustic encoders. (Acoustic features increase recognition of sentence breaks etc.)

-Learn contextual representations through unsupervised learning, since labelled data is limited.

-Explores using attention mechanism to learn alignment of lexical and acoustic features.

-Demonstrate adaptation to streaming usecase by limiting future context

-Effect of pretrained encoders on varying data sizes

-Use of N-best lists from ASR to perform data augmentation and improve performance on ASR outputs.

Approaches to Multimodal fusion alignment:

1. Forced alignment fusion

Uses LSTM-based acoustic encoder which uses force-aligned word durations to obtain word boundaries. The boundaries are used to select the LSTM state outputs to form word-level features. This output is assigned to all subwords of the same word.

1. Attention fusion

Force-aligned durations may limit acoustic context to few frames, thus introduce attention module with scaled dot-product attention to obtain alignment. Downsample input sequence with stride=2 in 1D Conv layer

Improve Robustness to ASR errors:

-Data Augmentation using ASR outputs for training. Use edit distance measure to align ASR hypothesis with reference punctuated text. Shown to improve scores with 3-best lists giving the best performance.

Result:

-pretrained BERT outperforms BLSTM by ~ 2%

-Proposed model outperforms lexical BERT but is still lacking when tested on ASR outputs.

-Acoustic information helped to improve punctuation across {.,?}.

Comments:

Output text might contain unwanted full-stops or ellipsis or commas when there the pause between words in someone’s speech is too long (i.e. for speech to text keyboard), where lexical version would be more accurate.

## [Dice Loss for Data-imbalanced NLP Tasks (arxiv.org)](https://arxiv.org/pdf/1911.02855.pdf)

29 Aug 2020

### Overview: Shows how Dice Loss performs better for tasks with imbalanced classes where F1 score is more relevant than accuracy as compared to Cross Entropy Loss.

1. Tries to handle class imbalance using losses based on the Sørensen–Dice coefficient (Sorensen, 1948) or Tversky index (Tversky, 1977). Sørensen–Dice coefficient (dice loss) is the harmonic mean of precision and recall. It attaches equal importance to false positives (FPs) and false negatives (FNs) and is thus more immune to data-imbalanced datasets. Tversky index extends dice loss by using a weight that trades precision and recall, which can be thought as the approximation of the Fβ score, and thus comes with more flexibility.

Dice Loss: [2pi · yi + γ]/[pi + yi + γ] or [2pi · yi + γ]/[pi2 + yi2 + γ]

1. Use Dynamic weight adjusting strategy, associating each training example with a dynamically changing weight proportionate to 1-p, allowing model to be more attentive to hard negative examples. Similar to idea of focal loss.

[2(1 – pi)α pi · yi + γ]/[(1 − pi)α pi + yi + γ]

Existing alternative approaches:

Resampling: oversampling weaker classes, boosting algorithms select harder examples, or controlling weight of examples during training (FocalLoss) which emphasizes harder examples while training.

Demonstration with Binary Classification task

Cross Entropy (CE) : -1/N (sum yij pij over example i and class j) ###Double check definition

Weighted CE: -1/N (sum (ai \* sum yij pij over class j) over example i)

Sørensen–Dice coefficient: 2|A ∩ B| / |A| + |B|

Result: DSC loss > Dice Loss > Focal Loss > BERT tagger baseline > Word Char Lattice LSTM > Joint POS

Results consistent across multiple tasks like NER or POS Tagging or Machine Reading Comprehension

Performed experiment on dataset of varying proportions (i.e. positive / negative augmentation) and produced similar results.

Did not perform as well as cross-entropy loss on accuracy-oriented task like SST dataset.

* Is the new BERT BLSTM CRF able to classify more than the standard 4 punctuation (? . ! ,)? Hypothesis: since BERT itself is trained on data with all punctuation classes, if there is a way to deal with imbalanced classes, it might be able to perform well. It is useful as they provide additional meaning to the data. Assuming the punctuation in the training data is accurate.
* Would performing task specific pretraining using wiki dataset improve accuracy on specific dataset?
* Random Insertion deletion, or substitution with synonyms, would homophone substitution be effective?
  + NLPAug
* Introduce examples which are shorter to allow transference to text with less context.
* Tried Dice Loss with weights proportionate to distribution, outcome: output has too little of the highest class. Weights too extreme
* Tried self-adjusting loss seems to fail, keeps increasing to 1? Maybe implementation wrong or lr too big?

Forgot to zero gradients

* Possibility of predicting double punctuations? Maybe pass the text through the model once, add the generated punctuations as new tokens, shift by half its length, and pass the text through the model a second time? Reduce the stride to accommodate new punctuations?
* Ability of model trained on longer sequence to generalize to shorter sequences?
* Try just the generic 4 punctuations , . ? ! first? I rather not actually if the model can actually learn the less common punctuation.
* TODO: Create Recall and Precision metrics and output training stuff to tensorboard. Shift code from colab to the cloud shell / github.
* CRF vs Dice loss? What exactly is CRF…
* How is Dice loss optimised?
* [Training Conditional Random Fields for Maximum Labelwise Accuracy (stanford.edu)](http://ai.stanford.edu/~olga/papers/nips2006-CRFtraining.pdf)?