Natural Language Processing

Lecture 17: Contextualized Embeddings
Classification

Set of documents → Topic
Preprocessing documents
Train/dev/test splits
Choose embedding options
Choose model architecture
  • Hyperparameters
  • Train and check it converges
  • Do experiments
• Error analysis
• Check with test set
The task

• Set of news articles plus topic
  • A few paragraphs (well written text)
  • 1 of 5: Business, Politics, Sport, Arts, Tech
• How much data?
• Is it “fair” data?
• Is it a “predictable” task
  • Too easy or too hard
Preprocessing

• Into some normal form
  • Remove formatting?
  • Upper/Lower case,
  • Remove numbers/dates/names?
• Word embedding with further tokenize
• How will your (ultimate) test data be different
Class distribution

• What is the class distribution
  • Is it balanced (near balanced)?
  • Should you merge/ignore any classes?
  • Should you exclude data
  • Should you resample data
Train/dev/test split

• (Maybe) 80% 10% 10% split
  • First 80%, next 10%, next 10%
  • Select batches throughout data
• Other restrictions
  • (News) separate days/weeks
  • Separate authors, subjects
  • Does dev & test reflect ultimate test set
• Not trying to get test set with highest score
• Trying to get highest number on test set
Choose your embedding technique

- word2vec/Glove/BERT/train your own
- Does your data match pretraining data
- Do you have special treatment of tokens
  - Local meanings (e.g. place/people names)
- Is it all English
- Other BERT-like models?
  - xln-roberta, mbert
- Size of embedding
- Document vs word embeddings
Choose your architecture

• Feed forward, CNN, LSTM, other
• Loss function: perplexity, accuracy, F-score, ...
• Fixed embeddings or varying
• Hyperparameters:
  • vector sizes, dropout, SGD, batch size
• Higher level things:
  • GANs, Reinforcement Learning
• Try simpler things first, then more complex things
Train and Test

• Train your model
• Test on dev set (not test set)
• Does it converge? (how many epochs)
• Does it predict?
  • Just majority class? Good distribution?
• Once you have a good(ish) model, experiment
• Do comparisons that should be better/worse
  • Are they better/worse?
Error Analysis

• Error measure appropriateness
  • Accuracy vs F-score (class equality)
  • Precision vs recall
• Use confusion matrix
  • What is getting confused, what works well
• Where is good/bad, why?
  • Look at false positives/negatives
  • Look at correct ones, are they good/easy?
• What is missing
• What extra information would you use to get it right
Error Analysis (High Level)

• What is the cost of errors?
  • Precision vs recall
  • Over prediction vs under prediction
• Biases?
  • All data/models/training are biased
  • Can you fix reduce that bias
• Is prediction based on naive reasons?
  • People called “Donald” are president
  • People called “Elon” are billionaires
Other Techniques

• Get more training data
• Data augmentation
  • Adding systemically modified examples
  • Generating new “similar” examples (e.g. MT)
• What is the best you can ever get?
  • How close are you to that
Getting Run-time examples

- https://huggingface.co
  - Code and tutorials
  - Pytorch and Tensorflow
- Other on-line tutorials
  - Step by step examples, try them!
- General code examples may be good (or not)
  - Make their code work for you before changing it
I need more GPUs

• Google co-lab gives access
• You probably don’t need more GPUs
  • You need a better model/dataset
• In NLP (and elsewhere)
  • You are graded on the task success
  • You aren’t graded on your global warming contributions
• (But sometimes you do need more GPUs)
  • But you need to justify it
Now run your model on your test set
  • You thought I’d forgotten about that
Run at least three models on your test set
  • Do models perform on your test set as with your dev set?
Get a new test set from another source
  • Does your models still work?
And Finally

• Know your data
• Know your models
• If you are surprised by a result
  • Investigate it (there is a bug, or you are brilliant)
• Stupid train/test data mismatch → biggest errors
• Tiny changes in conditions shouldn’t break models