Evaluation in NLP

CS 4740 (and crosslists): Introduction to Natural Language Processing
https://courses.cs.cornell.edu/cs4740/2025sp

Slides developed by:
Magd Bayoumi, Claire Cardie, Tanya Goyal, Dan Jurafsky, Lillian Lee, James Martin, Marten van Schijndel

Announcements

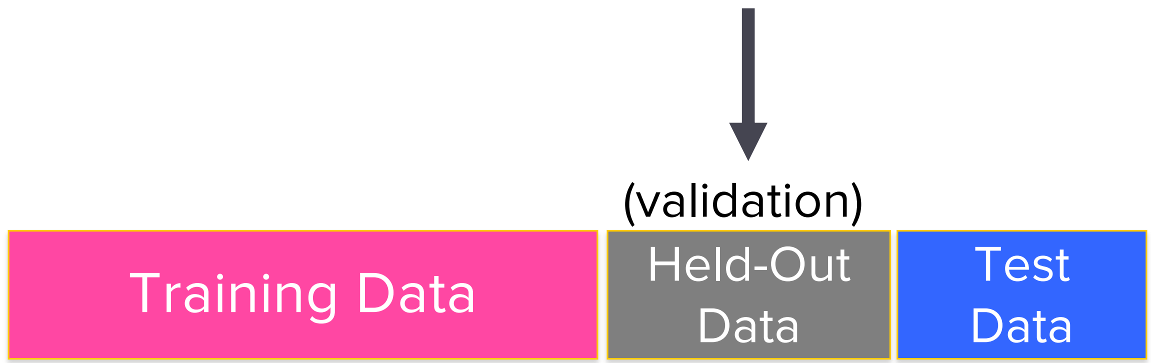
Today: How do we evaluate the performance of NLP systems?

More generally, how do we evaluate ML models?

Train **parameters** of our model on a **training set/training corpus**.

Test the model's performance on data we haven't seen: the **test set**

- No overlap with the training set.
- The **evaluation metric** tells us how well our model performs on the test set.



1. Train the model on the **training data**
2. Select an appropriate **evaluation metric**
3. Choose model + hyperparameters to maximize performance on held-out/**validation data**
4. Use selected model+hyperparameters when applying model to test set
5. Report results on **the test set**

Today: How do we evaluate the performance of NLP systems?

- Text classification
- LMs
- Sequence tagging
    - POS tagging
    - NER tagging

Evaluating text classifiers

Let's concentrate on binary classifiers:

- Is this email spam?

  spam (+) or not spam (-)

- Does this post have positive or negative sentiment?

  positive sentiment(+) or negative sentiment (-)

We'll need to know

1. What did our classifier say about each email or post?
2. What should our classifier have said, i.e., the correct answer, usually as defined by humans ("**gold label**")

First step in evaluation: The confusion matrix

		gold standard labels	
		gold positive	gold negative
system output labels	system positive	**true positive**	**false positive**
	system negative	**false negative**	**true negative**

Accuracy on the confusion matrix

gold standard labels

system output labels		gold positive	gold negative
	system positive	**true positive**	**false positive**
	system negative	**false negative**	**true negative**

correct

$$\textbf{accuracy} = \frac{\text{tp+tn}}{\text{tp+fp+tn+fn}}$$

total # of examples

Why don't we use accuracy?

Accuracy doesn't work well when we're dealing with uncommon or imbalanced classes

Suppose we look at 1,000,000 emails for spam

- 100 of them are spam
- 999,900 are not spam

Imagine the following simple classifier

Every post is not spam

Accuracy re: spam

100 posts are +; 999,900 are -

		gold standard labels	
		gold positive	gold negative
system output labels	system positive	**true positive**	**false positive**
	system negative	**false negative**	**true negative**

$$\textbf{accuracy} = \frac{\text{tp+tn}}{\text{tp+fp+tn+fn}}$$

Why don't we use accuracy?

Accuracy of our ”all posts are negative" classifier

- 999,900 true negatives and 100 false negatives
- Accuracy is 999,900/1,000,000 = 99.99%!
- But useless at finding spam!!
- Which was our goal!

Accuracy doesn't work well for unbalanced classes

- Most emails are not spam!

Instead of accuracy we use precision and recall

gold standard labels

system output labels		gold positive	gold negative	
	system positive	**true positive**	**false positive**	**precision** $= \frac{\text{tp}}{\text{tp+fp}}$
	system negative	**false negative**	**true negative**	
		recall $= \frac{\text{tp}}{\text{tp+fn}}$		**accuracy** $= \frac{\text{tp+tn}}{\text{tp+fp+tn+fn}}$

Precision: % of *selected* items (i.e. identified as positive class) that are correct

Recall: % of *targeted* items (i.e. gold positives) that that are correct

Precision/Recall aren't fooled by the "just call everything spam" classifier!

Stupid classifier

- Accuracy = 999,900/1,000,000 = 99.99%

But the Recall and Precision for this classifier are terrible:

$$\textbf{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

$$\textbf{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

A combined measure: F-measure

F-measure usually refers to F1 --- a combination (the harmonic mean) of precision and recall.

$$\mathrm{F}_1 = \frac{2PR}{P+R}$$

Today: Evaluating NLP systems

- Text classification
- **LMs**
- Sequence tagging
    - POS tagging
    - NER tagging

How do we evaluate a language model?

unigram, bigram, trigram counts …

Train **parameters** of our model on a **training set/training corpus**.

Select an **evaluation metric**

Choose model + hyperparameters to maximize the probability of held-out/**validation data**

Test the model's performance on data we haven't seen: the **test set**

How do we determine when a language model is performing well?

Does our model assign higher probability to “real” / grammatical sentences?

Best option: Extrinsic evaluation of model

- Embed LM in an application / task
  - Machine translation, Autocomplete, Speech recognition
- Measure the performance **on the application/task** with and without the LM using an **evaluation metric designed for that task**
- Compare performance of model A and B

Best option: Extrinsic evaluation of LM

- Embed LM in an application / task
  - Machine translation, Autocomplete, Speech recognition
- Measure LM performance **on the application/task** with and without the LM using an **evaluation metric designed for that task**
  - How many words translated correctly?
  - How many future words are predicted correctly?
  - How many words are transcribed correctly?
  - …
- Compare performance of model A and B

Difficulty of **extrinsic** evaluation

- Time consuming
  - Can take days or weeks to obtain “gold standard” / labelled / annotated test data

Alternative for LM evaluation: **intrinsic** evaluation

- Measure quality of the LM model independent of any application
- Standard measure: **perplexity**
  - Intuition: the better model is the one that has a tighter fit to the test data….one that predicts the test data

- Perplexed == confused

- We'll look first at the measure
    - Lower is better
- Then on how to correctly apply it

Intuition behind Perplexity

Shannon game

- Similar to the predict next word task

I always order burgers with _____

The cat sat on the _____

NLP is _____

- Unigrams are quite bad at fitting the test data
    - Why?

fries 0.1
ketchup 0.1
onions 0.1
lettuce 0.1
tomato 0.1
cheese 0.1
…
cheetos 0.0001
…
the 1e-1000

Perplexity (PP)

For a test set $W = w_1 w_2 \ldots w_N$

$$PP(W) = P(w_1 w_2 \ldots w_N)^{-1/N}$$

$$= \sqrt[N]{\frac{1}{P(w_1 w_2 \ldots w_N)}}$$

The higher the (estimated) probability of the word sequence, the **lower** the perplexity.

Must be **computed with models that have no knowledge of the test set**, i.e. were not trained on (any part of) the test corpus.

Perplexity

For a test set $W = w_1 w_2 \dots w_N$

$$PP(W) = P(w_1 w_2 \dots w_N)^{-\frac{1}{N}}$$

$$= \sqrt[N]{\frac{1}{P(w_1 w_2 \dots w_N)}}$$

From earlier lectures...

use chain rule

$$= \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i \mid w_1 \dots w_{n-1})}}$$

assume bigram approx.

$$= \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i \mid w_{i-1})}}$$

Perplexity as weighted branching factor

- Branching factor - the number of possible next words that can follow any word
- Consider random sequence of digits (0, …, 9)
  - Possible branches….?

Perplexity as weighted branching factor

- Branching factor - the number of possible next words that can follow any word
- Consider random sequence of digits (0, ..., 9)
  - Possible branches....?

$$
\begin{aligned}
PP(W) &= P(w_1 w_2 \ldots w_n)^{-\frac{1}{N}} \\
&= (\frac{1}{10}^N)^{-\frac{1}{N}} \\
&= \frac{1}{10}^{-1} \\
&= 10
\end{aligned}
$$

Lower perplexity = better model

Training: 38 million words, WSJ

Test set: 1.5 million words, a different portion of WSJ

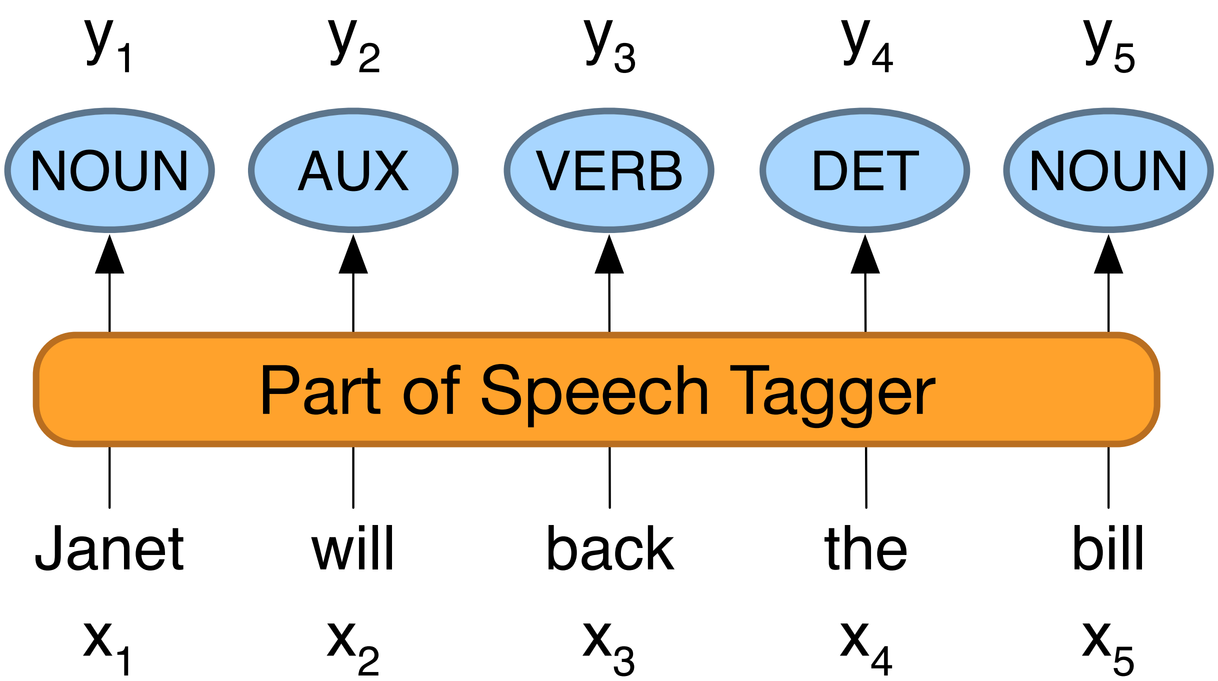
	Unigram	Bigram	Trigram
Perplexity	962	170	109

Today: Evaluating NLP systems

- Text classification
- LMs
- Sequence tagging
  - POS tagging
  - NER tagging

Part-of-Speech Tagging

Map from sequence $x_1,\ldots,x_n$ of words to $y_1,\ldots,y_n$ of POS tags



Sample "Tagged" English sentences

Preliminary/ADJ findings/NOUN were/AUX reported/VERB in/ADP today/NOUN ’s/PART New/PROPN England/PROPN Journal/PROPN of/ADP Medicine/PROPN

Which evaluation metric should we use?

accuracy

NER

[PER Jane Villanueva] of [ORG United] , a unit of [ORG United Airlines Holding] , said the fare applies to the [LOC Chicago] route.

Accomplished via BIO tagging

Which evaluation metric should we use?

Words	**BIO Label**
Jane	B-PER
Villanueva	I-PER
of	O
United	B-ORG
Airlines	I-ORG
Holding	I-ORG
discussed	O
the	O
Chicago	B-LOC
route	O
.	O

NER

[PER Jane Villanueva] of [ORG United] , a unit of [ORG United Airlines Holding] , said the fare applies to the [LOC Chicago] route.

- Not accuracy!!! Why not?
  - Segmentation component of the task causes problems
- **Entity** is the correct unit to measure
- Use R/P/F

recall = # correctly identified NEs / total # of NEs that should have been identified

precision = # correctly identified NEs / # of NEs that were identified

("correct" according to *exact match* or *partial match* or *proportional match)*

Note: There remains a mismatch between the training conditions (token-level) and the testing conditions (entity-level)