Lecture 4-2: Language Models

How can we distinguish word salad, spelling errors and grammatical sentences?

> `Language models` define probability distributions over the strings in a language.

> `N-gram models` are the simplest and most common kind of language model.

Why do we need language models?

Many NLP tasks require natural language output:

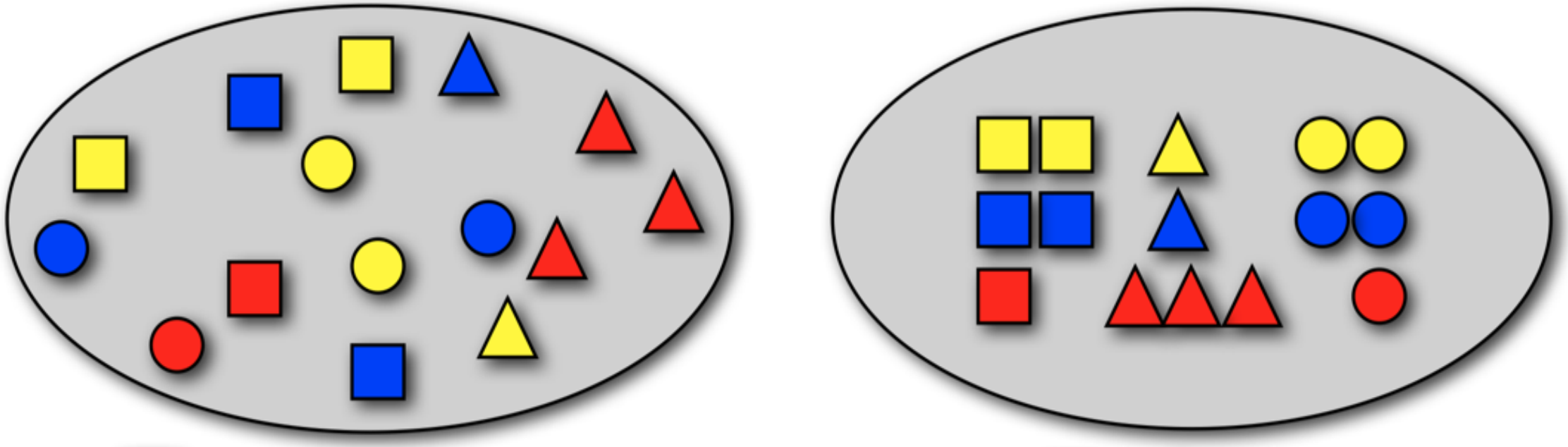
- `Machine translation` : return text in the target language
- `Speech recognition` : return a transcript of what was spoken
- `Natural language generation (NLU)` : return natural language text
- `Spell-checking` : return corrected spelling of input

Language models define `probability distributions over (natural language) strings or sentences` .

- We can use a language model to generate strings
- We can use a language model to score/rank candidate strings so that we can choose the best (i.e. most likely) one:
  if $P_{LM}(A) > P_{LM}(B)$, return A, not B

Sampling with replacement

Pick a random shape, then put it back in the bag.



$P(\blacksquare) = 2/15$ $P(\blacksquare) = 1/15$ $P(\blacksquare \text{ or } \blacktriangle) = 2/15$

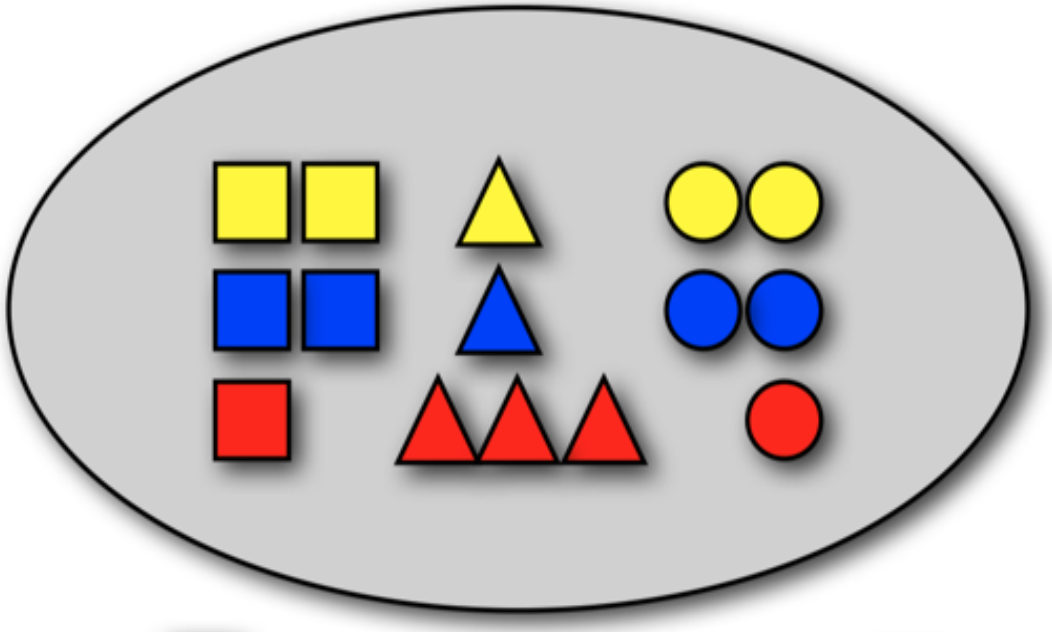
$P(\text{blue}) = 5/15$ $P(\text{red}) = 5/15$ $P(\triangle \mid \text{red}) = 3/5$

$P(\text{blue} \mid \square) = 2/5$ $P(\square) = 5/15$

Sampling with replacement

Pick a random shape, then put it back in the bag.

What sequence of shapes will you draw?



$P(\bigcirc\triangle\triangle\square) = 1/15 \times 1/15 \times 1/15 \times 2/15$

$= 2/50625$

$P(\triangle\bigcirc\bigcirc\triangle) = 3/15 \times 2/15 \times 2/15 \times 3/15$

$= 36/50625$

$P(\square) = 2/15$ $P(\square) = 1/15$ $P(\square \text{ or } \triangle) = 2/15$

$P(\text{blue}) = 5/15$ $P(\text{red}) = 5/15$ $P(\triangle \mid \text{red}) = 3/5$

$P(\text{blue} \mid \square) = 2/5$ $P(\square) = 5/15$

Now let's look at natural language

Text as a bag of words

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, 'and what is the use of a book,' thought Alice 'without pictures or conversation?'

$P(\text{of}) = 3/66$
$P(\text{Alice}) = 2/66$
$P(\text{was}) = 2/66$
$P(\text{to}) = 2/66$
$P(\text{her}) = 2/66$
$P(\text{sister}) = 2/66$
$P(\text{,}) = 4/66$
$P(\text{'}) = 4/66$

> In this model, P(English sentence) = P(word salad)

Probability theory: terminology

Trial (aka "experiment")

- Picking a shape, predicting a word

Sample space Ω:

- The set of all possible outcomes (all shapes; all words in Alice in Wonderland)

Event $\omega \subseteq \Omega$:

- An actual outcome (a subset of Ω) (predicting 'the', picking a triangle)

Random variable $X : \Omega \to T$

- A function from the sample space (often the identity function)
- Provides a 'measurement of interest' from a trial/experiment
  (Did we pick 'Alice'/a noun/a word starting with "x"/...? How often does the word 'Alice' occur? How many words occur in each sentence?)

Discrete probability distributions: Single Trials

`Discrete` : a fixed (often finite) number of outcomes

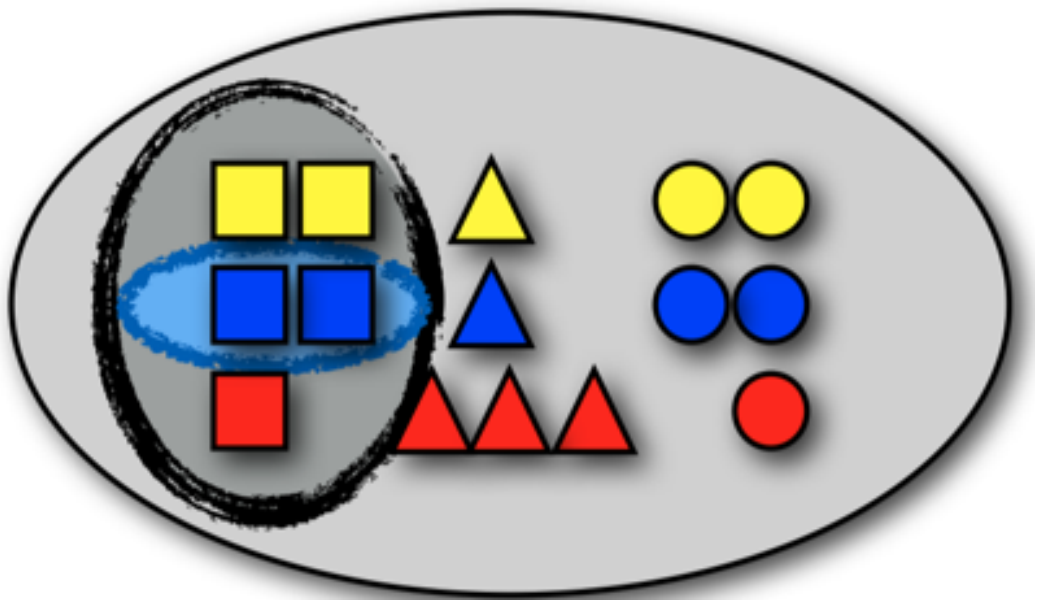
- `Bernoulli distribution` (Two possible outcomes (head, tail)
  Defined by the probability of success (= head/yes)
  The probability of head is p. The probability of tail is $1 - p$.
- `Categorical distribution` (N possible outcomes c1 ...cN)
  The probability of category/outcome c_i is $p_i (0 \leq p_i \leq 1; \sum_{p_i} = 1)$.
  e.g.) the probability of getting a six when rolling a die once
  e.g.) the probability of the next word (picked among a vocabulary of N words)
  (NB: Most of the distributions we will see in this class are categorical.
  Some people call them multinomial distributions, but those refer to sequences of trials, e.g.) the probability of getting five sixes when rolling a die ten times)

Joint and Conditional Probability

The `conditional probability` of X given Y, $P(X|Y)$, is defined in terms of the `probability` of Y, $P(Y)$, and the joint probability of X and Y, $P(X, Y)$:

$$P(X|Y) = \frac{P(X, Y)}{P(Y)}$$

What is the probability that we get a blue shape if we pick a square?



The chain rule

The `joint probability` $P(X, Y)$ can also be expressed in terms of the `conditional probability` $P(X|Y)$

$$P(X, Y) = P(X|Y)P(Y)$$

Generalizing this to `N` joint events (or random variables) leads to the so-called `chain rule`:

$$P(X_1, X_2, \ldots, X_n) = P(X_1)P(X_2|X_1)P(X_3|X_2, X_1)\ldots P(X_n|X_1, \ldots X_{n-1})$$
$$= P(X_1)\prod_{i=2}^{2} P(X_i|X_1 \ldots X_{i-1})$$

Independence

Two events or random variables X and Y are `independent` if

$$P(X, Y) = P(X)P(Y)$$

If X and Y are independent, then $P(X|Y) = P(X)$:

$$\begin{aligned} P(X|Y) &= \frac{P(X, Y)}{P(Y)} \\ &= \frac{P(X)P(Y)}{P(Y)} \\ &= P(X) \end{aligned}$$

Probability models

Building a probability model consists of two steps:

1. Defining the model
2. Estimating the model's parameters (= training/learning)

Probability models (almost) always make `independence assumptions`.

- Even though X and Y are not actually independent, our model may treat them as independent.
- This can drastically reduce the number of parameters to estimate.
- Models without independence assumptions have (way) too many parameters to estimate reliably from the data we have
- But since independence assumptions are often incorrect, those models are often incorrect as well: they assign probability mass to events that cannot occur

Language modeling with N-grams

A `language model` over a vocabulary V assigns probabilities to strings drawn from V^*.

How do we compute the probability of a string $w^{(1)} \dots w^{(i)}$?

Recall the `chain rule`:

$$P(w^{(1)} \dots w^{(i)}) = P(w^{(1)}) \cdot P(w^{(2)}|w^{(1)}) \cdot \dots \cdot P(w^{(i)}|w^{(i-1)}, \dots, w^{(1)})$$

An `n-gram` language model assumes each word depends only on `the last n-1 words`:

$$P_{ngram}(w^{(1)} \dots w^{(i)}) = P(w^{(1)}) \cdot P(w^{(2)}|w^{(1)}) \cdot \dots \cdot P(w^{(i)}|w^{(i-1)}, \dots, w^{(1-(n+1))})$$

N-gram models

N-gram models assume each word (event) `depends only on the previous n–1` words (events):

$$\text{Unigram model: } P(w^{(1)} \dots w^{(i)}) = \prod_{i=1}^{N} P(w^{(i)})$$

$$\text{Bigram model: } P(w^{(1)} \dots w^{(i)}) = \prod_{i=1}^{N} P(w^{(i)} | w^{(i-1)})$$

$$\text{Trigram model: } P(w^{(1)} \dots w^{(i)}) = \prod_{i=1}^{N} P(w^{(i)} | w^{(i-1)}, w^{(i-2)})$$

- Independence assumptions where the n-th event in a sequence depends only on the last n-1 events are called Markov assumptions (of order n–1).

How many parameters do n-gram models have?

Given a vocabulary V of $|V|$ word types: for $|V| = 10^4$:

> `Unigram model`: $|V|$ parameters, 10^4 parameters
> (one distribution $P(w^{(i)})$ with $|V|$ outcomes [each $w \in V$ is one outcome])

> `Bigram model`: $|V|^2$ parameters, 10^8 parameters
> ($|V|$ distributions $P(w^{(i)}|w^{(i-1)})$, one distribution fore each $w \in V$ with $|V|$ outcomes [each $w \in V$ is one outcome])

> `Trigram model`: $|V|^3$ parameters, 10^{12} parameters
> ($|V|^2$ distributions $P(w^{(i)}|w^{(i-1)}, w^{(i-2)})$, one distribution fore each bigram $w'w''$ with $|V|$ outcomes [each $w \in V$ is one outcome])

Sampling with replacement

beginning by, very Alice but was and?
reading no tired of to into sitting
sister the, bank, and thought of without
her nothing: having conversations Alice
once do or on she it get the book her had
peeped was conversation it pictures or
sister in, 'what is the use had twice of
a book''pictures or' to

$P(\texttt{of}) = 3/66$ $P(\texttt{to}) = 2/66$ $P(\texttt{,}) = 4/66$

$P(\texttt{Alice}) = 2/66$ $P(\texttt{her}) = 2/66$ $P(\texttt{'}) = 4/66$

$P(\texttt{was}) = 2/66$ $P(\texttt{sister}) = 2/66$

In this model, $P(\textit{English sentence}) = P(\textit{word salad})$

A bigram model for Alice

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, 'and what is the use of a book,' thought Alice 'without pictures or conversation?'

$P(w^{(i)} = \text{of} \mid w^{(i-1)} = \text{tired}) = 1$
$P(w^{(i)} = \text{of} \mid w^{(i-1)} = \text{use}) = 1$
$P(w^{(i)} = \text{sister} \mid w^{(i-1)} = \text{her}) = 1$
$P(w^{(i)} = \text{beginning} \mid w^{(i-1)} = \text{was}) = 1/2$
$P(w^{(i)} = \text{reading} \mid w^{(i-1)} = \text{was}) = 1/2$

$P(w^{(i)} = \text{bank} \mid w^{(i-1)} = \text{the}) = 1/3$
$P(w^{(i)} = \text{book} \mid w^{(i-1)} = \text{the}) = 1/3$
$P(w^{(i)} = \text{use} \mid w^{(i-1)} = \text{the}) = 1/3$

Using a bigram model for Alice

English

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, 'and what is the use of a book,' thought Alice 'without pictures or conversation?'

Word Salad

beginning by, very Alice but was and? reading no tired of to into sitting sister the, bank, and thought of without her nothing: having conversations Alice once do or on she it get the book her had peeped was conversation it pictures or sister in, 'what is the use had twice of a book''pictures or' to

Now, $P(English) \gg P(word\ salad)$

$P(w^{(i)} = \texttt{of} \mid w^{(i-1)} = \texttt{tired}) = 1$

$P(w^{(i)} = \texttt{of} \mid w^{(i-1)} = \texttt{use}) = 1$

$P(w^{(i)} = \texttt{sister} \mid w^{(i-1)} = \texttt{her}) = 1$

$P(w^{(i)} = \texttt{beginning} \mid w^{(i-1)} = \texttt{was}) = 1/2$

$P(w^{(i)} = \texttt{reading} \mid w^{(i-1)} = \texttt{was}) = 1/2$

$P(w^{(i)} = \texttt{bank} \mid w^{(i-1)} = \texttt{the}) = 1/3$

$P(w^{(i)} = \texttt{book} \mid w^{(i-1)} = \texttt{the}) = 1/3$

$P(w^{(i)} = \texttt{use} \mid w^{(i-1)} = \texttt{the}) = 1/3$

From n-gram probabilities to language models

A language $L \subseteq V^*$ is a (possibly infinite) set of strings over a (finite) vocabulary V.

$P(w^{(i)}|w^{(i-1)})$ defines a distribution over the words in V:

$$\forall w \in V : \left[\sum_{w' \in V} P(w^{(i)} = w'|w^{(i-1)} = w)\right] = 1$$

By multiplying this distribution N times, we get one distribution `over all strings of the same length` $N(V^N)$:

Prob. of one N-word string:

$$P(w_1 \dots w_2) = \prod_{i=1 \dots N} P(w^{(i)} = w_i|w^{(i-1)} = w_{i-1})$$

But instead of N separate distributions, we want one distribution over strings of any length

From n-gram probabilities to language models

We have just seen how to use n-gram probabilities to define one distribution $P(V^N)$ for each string length N.

But a `language model` $P(L) = P(^*)$ should define one distribution $P(V^*)$ that sums to one over all strings in $L \subseteq V^*$, regardless of their length:

$$P(L) = P(V) + P(V^2) + P(V^3) + \ldots + P(V^n) + \ldots = 1$$

Why do we want one distribution over L?

Why do we care about having one probability distribution for all lengths?

This allows us to compare the probabilities of strings of different lengths, because they're computed by the same distribution.

This allows us to generate strings of arbitrary length with one model.

How do we use language models?

Independently of any application, we could use a language model as a `random sentence generator`

- (we sample sentences according to their language model probability)

We can use a language model as a `sentences ranker`.

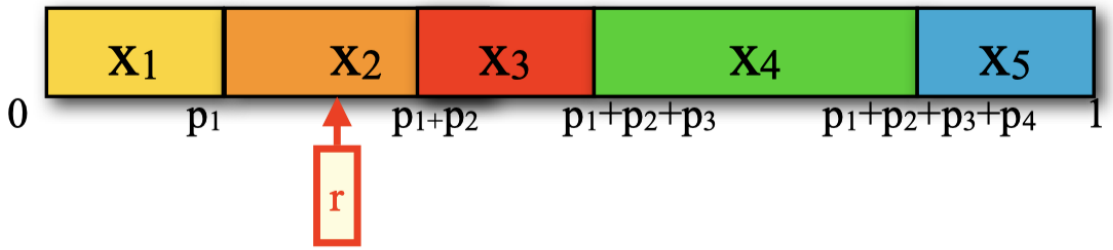
- Systems for applications such as machine translation, speech recognition, spell-checking, generation, etc. often produce many candidate sentences as output.
- We prefer output sentences S_{Out} that have a higher language model probability.

Generating from a distribution

How do you generate text from an n-gram model?

That is, how do you sample from a distribution P(X |Y=y)?

- Assume X has N possible outcomes (values): $\{x_1, \ldots, x_N\}$ and $P(X = x_i|Y = y) = p_i$
- Divide the interval [0,1] into N smaller intervals according to the probabilities of the outcomes
- Generate a random number r between 0 and 1.
- Return the x_1 whose interval the number is in.



Generating the Wall Street Journal

unigram: Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a acquire to six executives

bigram: Last December through the way to preserve the Hudson corporation N. B. E. C. Taylor would seem to complete the major central planners one point five percent of U. S. E. has already old M. X. corporation of living on information such as more frequently fishing to keep her

trigram: They also point to ninety nine point six billion dollars from two hundred four oh six three percent of the rates of interest stores as Mexico and Brazil on market conditions

Generating Shakespeare

Unigram	• To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have • Every enter now severally so, let • Hill he late speaks; or! a more to leg less first you enter • Are where exeunt and sighs have rise excellency took of.. Sleep knave we. near; vile like
Bigram	• What means, sir. I confess she? then all sorts, he is trim, captain. •Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow. •What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman? •Enter Menenius, if it so many good direction found'st thou art a strong upon command of fear not a liberal largess given away, Falstaff! Exeunt
Trigram	• Sweet prince, Falstaff shall die. Harry of Monmouth's grave. • This shall forbid it should be branded, if renown made it empty. • Indeed the duke; and had a very good friend. • Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.
Quadrigram	• King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in; • Will you not tell me who I am? • It cannot be but so. • Indeed the short and the long. Marry, 'tis a noble Lepidus.

Intrinsic vs Extrinsic Evaluation

How do we know whether one language model is better than another?

There are two ways to evaluate models:

- `intrinsic evaluation` measures how well the model captures what it is supposed to capture (e.g. probabilities)
- `extrinsic (task-based) evaluation` measures how useful the model is in a particular task.

Both cases require an `evaluation metric` that allows us to measure and compare the performance of different models.

Intrinsic evaluation

Define an `evaluation metric (scoring function)`.

- We will want to measure how similar the predictions of the model are to real text.

Train the model on a `'seen' training set`

- Perhaps: tune some parameters based on held-out data (disjoint from the training data, meant to emulate unseen data)

Test the model on an `unseen test set`

- (usually from the same source (e.g. WSJ) as the training data)
  Test data must be disjoint from training and held-out data
  Compare models by their scores (more on this in the next lecture).

Perplexity

The perplexity of a language models is defined as the inverse $\frac{1}{P(\dots)}$ of the probability of the test set, normalized $\sqrt[N]{\dots}$ by the # of tokens (N) in the test set.

If a LM assigns probability $P(w_1, \dots, w_N)$ to a test corpus $w_1 \dots w_N$, the LM's perplexity, $PP(w_1 \dots w_N)$, is

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

A LM with `lower perplexity is better` because it assigns `a higher probability to the unseen test corpus`.

Intrinsic vs. Extrinsic Evaluation

Perplexity tells us which LM assigns a higher probability to unseen text

This doesn't necessarily tell us which LM is better for our task (i.e. is better at scoring candidate sentences)

Task-based evaluation:

- Train model A, plug it into your system for performing task T
- Evaluate performance of system A on task T
- Train model B, plug it in, evaluate system B on same task T
- Compare scores of system A and system B on task T

Word Error Rate (WER)

Originally developed for speech recognition.

How much does the `predicted sequence of words differ from the actual sequence of words` in the correct transcript?

$$\text{WER} = \frac{\text{Insertions} + \text{Deletions} + \text{Substitutions}}{\text{Actual words in transcript}}$$

Insertions: "eat lunch" → "eat **a** lunch"

Deletions: "see **a** movie" → "see movie"

Substitutions: "drink **ice** tea" → "drink **nice** tea"