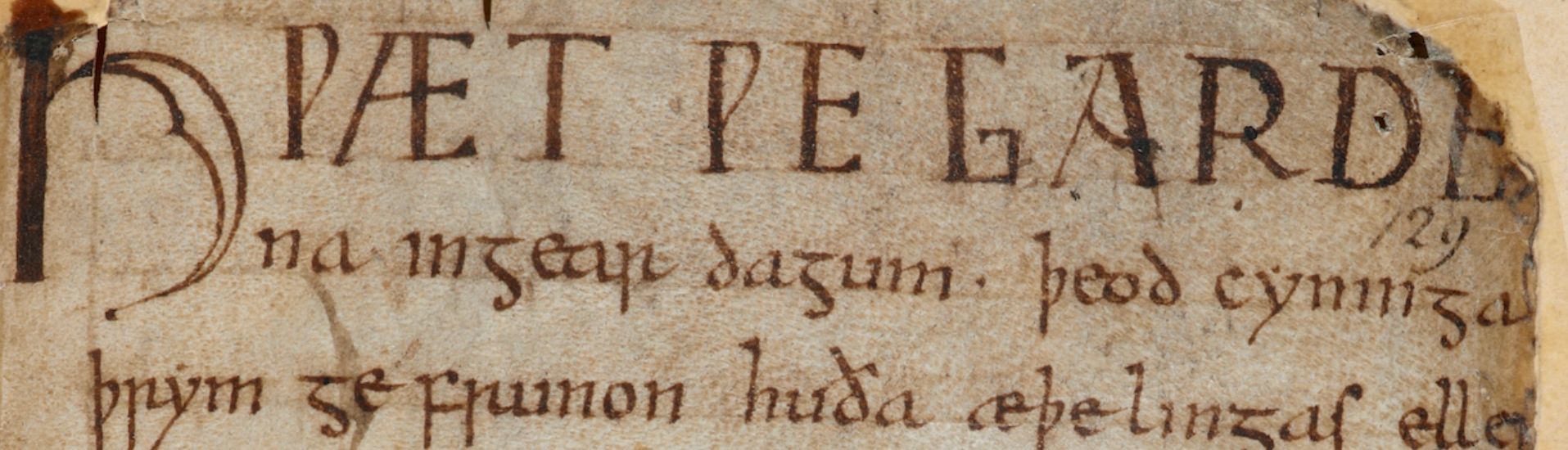




Natural Language Processing

Info 159/259

Lecture 6: Annotation (Feb 3, 2022)

David Bamman, UC Berkeley

Hwæt! Wé Gárde
na in géardagum, þéodcyninga
þrym gefrúnon, hú ðá æþelingas ellen

Natural Language Processing

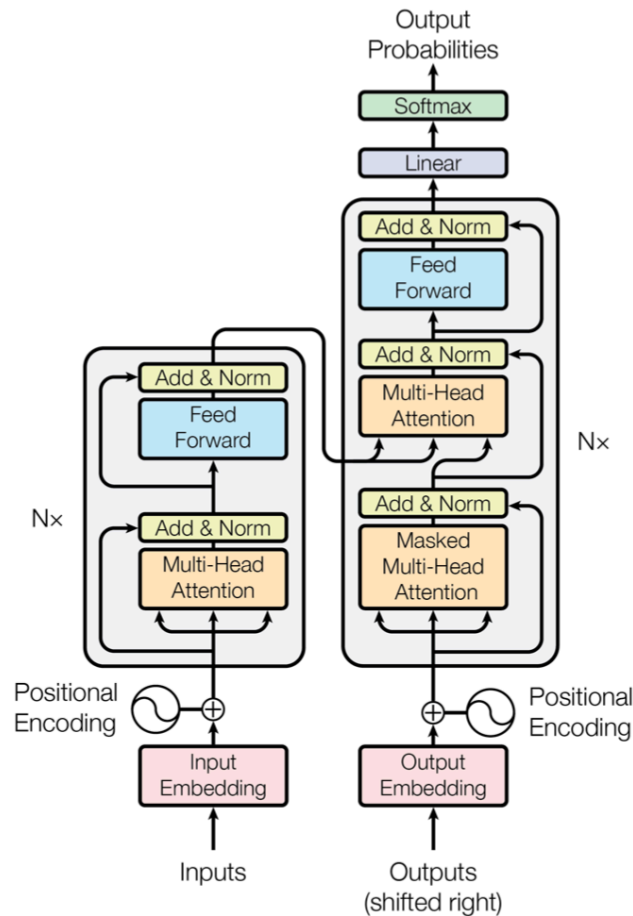
Info 159/259

Lecture 6: Annotation (Feb 3, 2022)

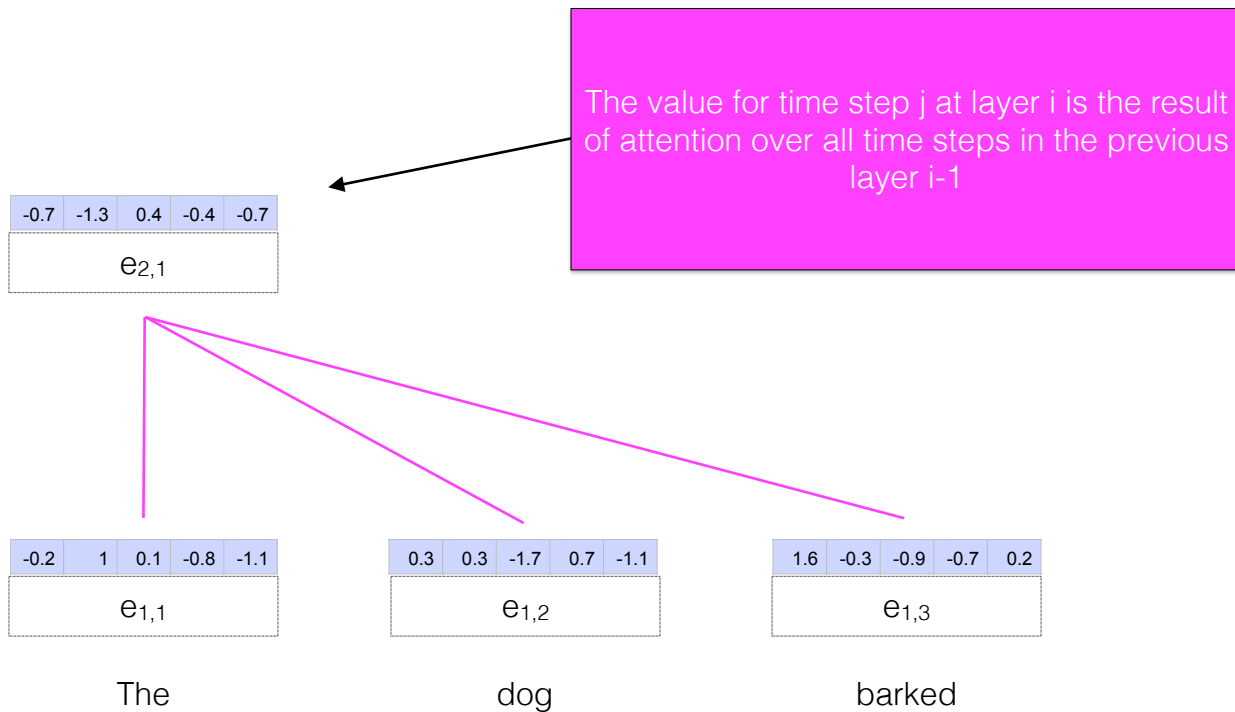
David Bamman, UC Berkeley

Transformers

- Vaswani et al. 2017, “Attention is All You Need”
- Transforms map an input **sequence** of vectors to an output **sequence** of vectors of the same dimensionality



Self-Attention



- Let's separate out the different functions that an input vector has in attention by transforming it into separate representations for its role in a weighted sum (the **value**) from the roles used to assess compatibility (the **query** and **key**).

query

$$q_{1,1} \in \mathbb{R}^{37} \quad (e_{1,1} W^Q)$$

key

$$k_{1,1} \in \mathbb{R}^{37} \quad (e_{1,1} W^K)$$

value

$$v_{1,1} \in \mathbb{R}^{100} \quad (e_{1,1} W^V)$$

original value

$$e_{1,1} \in \mathbb{R}^{100}$$



The

$$W^Q \in \mathbb{R}^{100 \times 37}$$

$$W^K \in \mathbb{R}^{100 \times 37}$$

$$W^V \in \mathbb{R}^{100 \times 100}$$

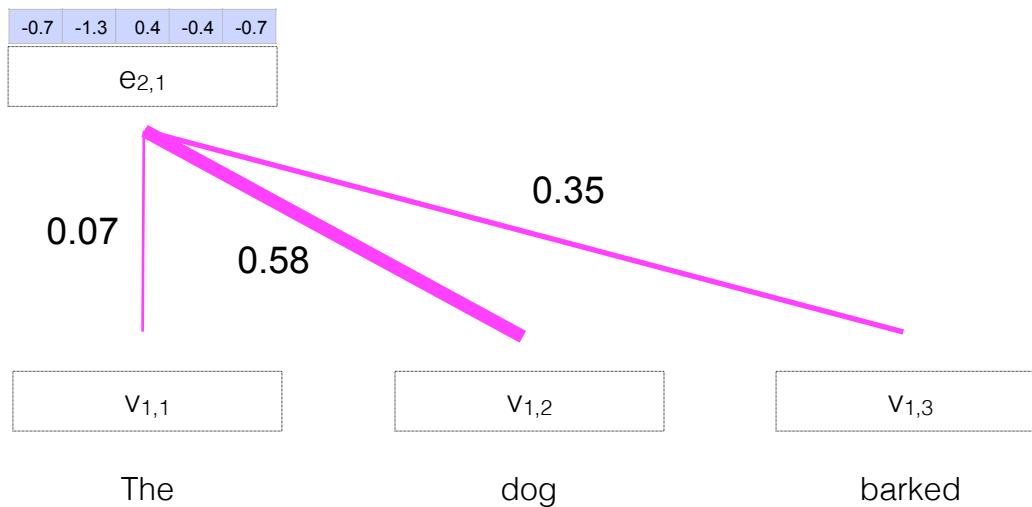
These are all parameters we *learn*. 100 is the original input dimension; 37 is a hyper-parameter we choose.

- The compatibility score between two words is the dot product between their respective **query** and **key** vectors.

$$\text{score}(e_i, e_j) = q_i \cdot k_j$$

<i>a</i>	0.07	0.58	0.35	$a = \text{softmax}(\text{scores})$
<i>scores</i>	-1.4	0.64	0.14	
	$q_1 \cdot k_1$	$q_1 \cdot k_2$	$q_1 \cdot k_3$	
	-0.210.1-0.8-1.1 $e_{1,1}$	0.30.3-1.70.7-1.1 $e_{1,2}$	1.6-0.3-0.9-0.70.2 $e_{1,3}$	
	The	dog	barked	

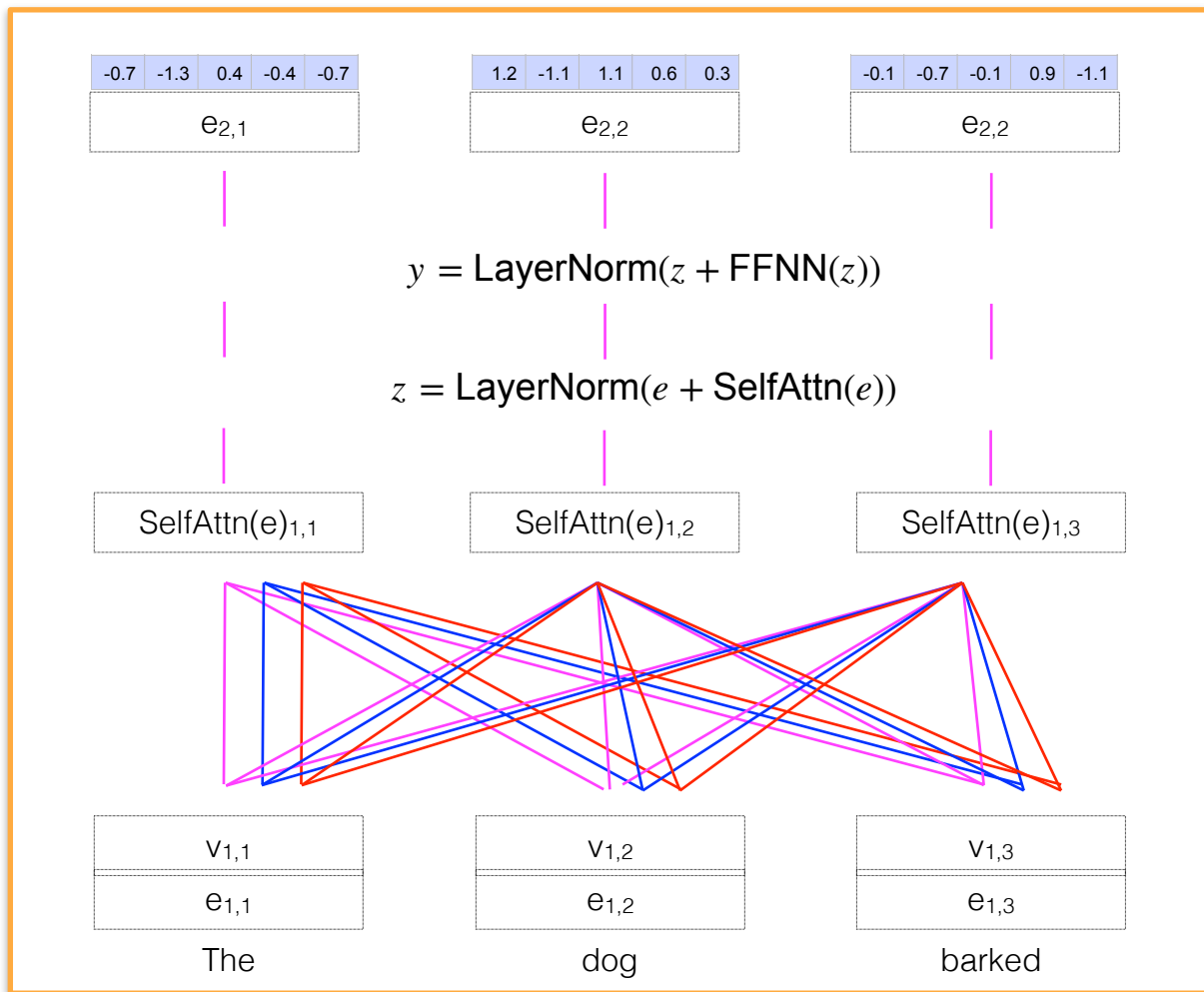
- The output of attention is a weighted sum over the **values** of the previous layer.



Output

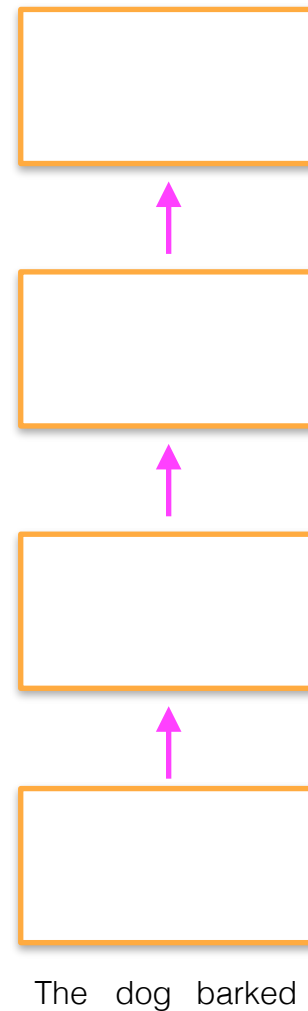
This whole process defines one attention **block**. The input is a sequence of (e.g. 100-dimensional) vectors; the output of each block is a sequence of (100-dimensional) vectors.

Input

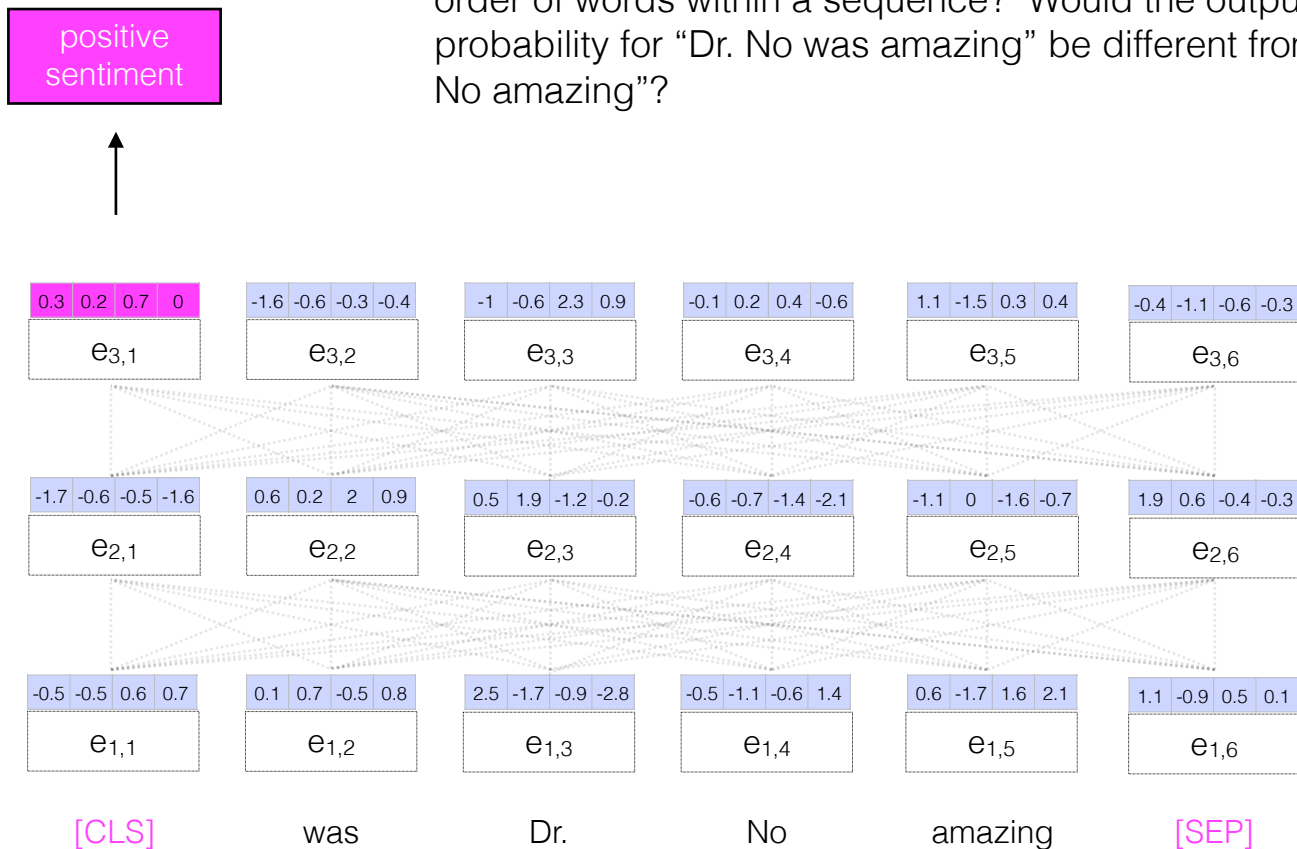


This whole process defines one attention **block**.
The input is a sequence of (e.g. 100-dimensional) vectors; the output of each block is a sequence of (100-dimensional) vectors.

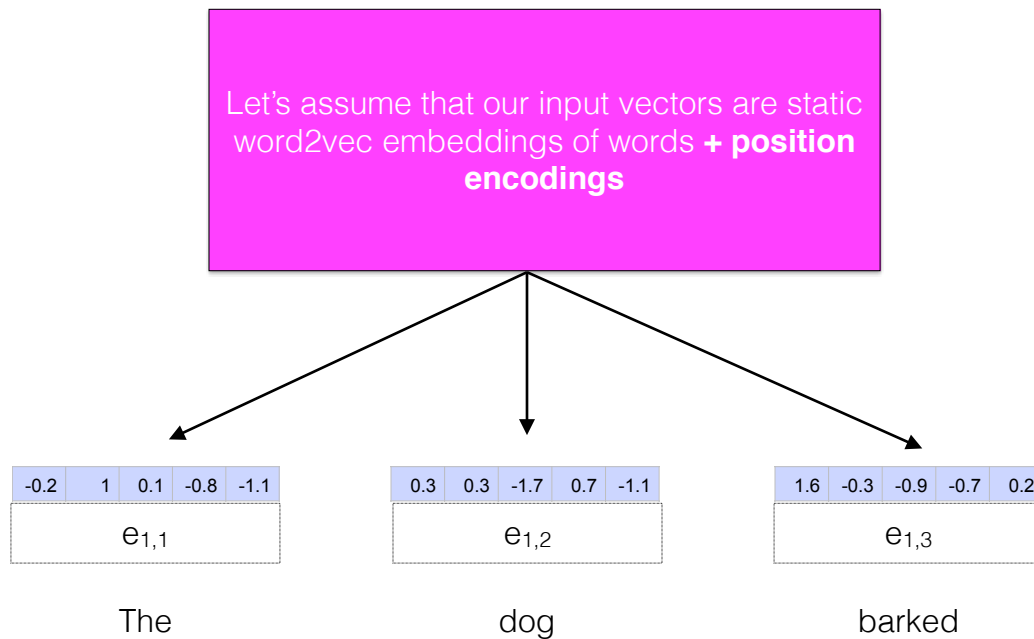
Transformers can stack many such blocks;
where the output from block b is the input to block $b+1$.



- Does a transformer encode any intrinsic information about the order of words within a sequence? Would the output probability for “Dr. No was amazing” be different from “was Dr. No amazing”?



Position encoding



Position embeddings

One option is to add learnable position embeddings $\text{pe}[i]$ to each word embedding e at position i (or concatenate them)

We can add two vectors if they're the same dimensionality

$$e_i = e_i + \text{pe}[i]$$

Or concatenate them if not

$$e_i = e_i \oplus \text{pe}[i]$$

0	2	-0.5	1.1	0.3	0.4	-0.5
1	-1.4	0.4	-0.2	-0.9	0.5	0.9
2	-1.1	-0.2	-0.5	0.2	-0.8	0
3	0.7	-0.3	1.5	-0.3	-0.4	0.1
4	-0.8	1.2	1	-0.7	-1	-0.4
5	0	0.3	-0.3	-0.9	0.2	1.4
6	0.8	0.8	-0.4	-1.4	1.2	-0.9
7	1.6	0.4	-1.1	0.7	0.1	1.6
...	...	...	...	...	...	...

position embeddings (pe)

Position encodings

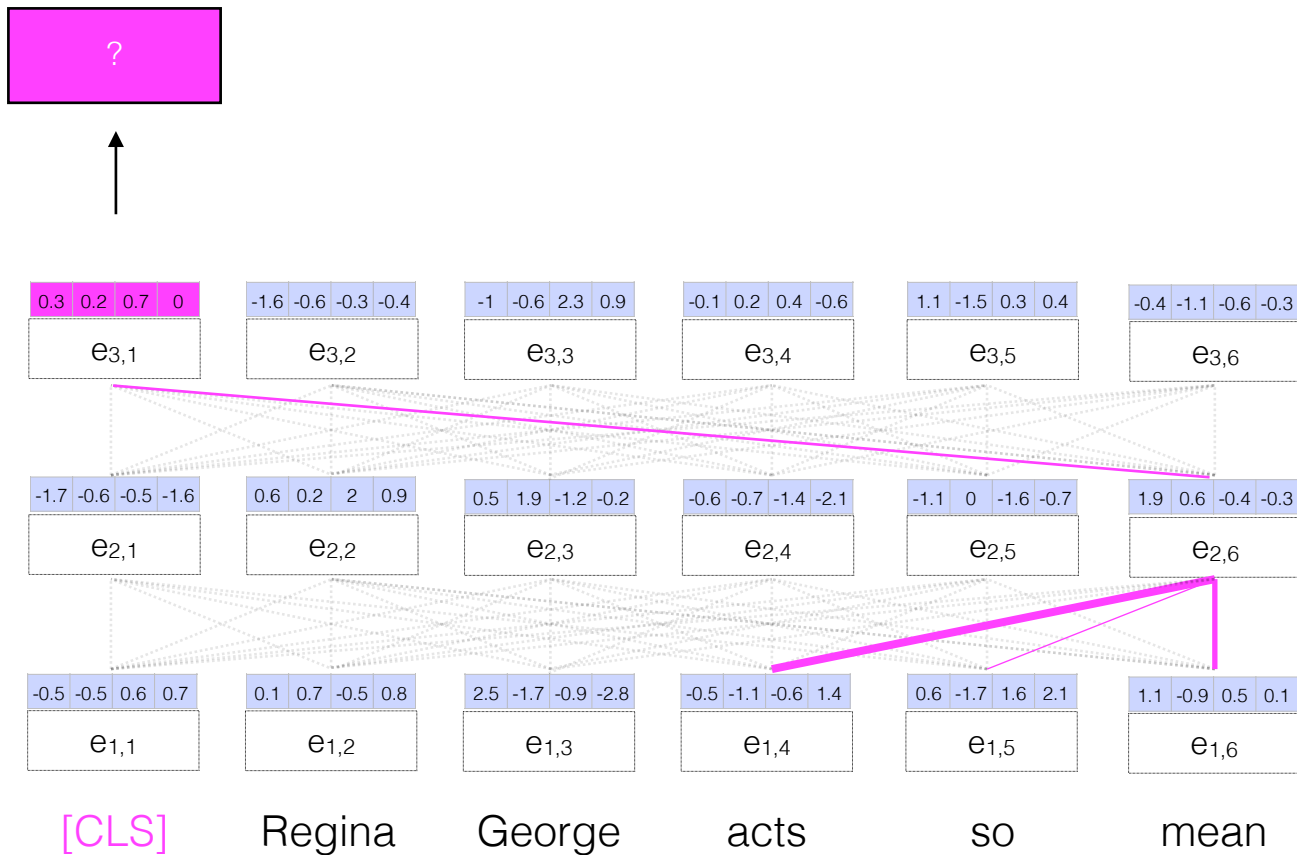
Vaswani et al. 2017 use sinusoidal functions to deterministically create a vector of position encodings the same dimensionality as the input (d_{model}); the embedding dimensions progress from wavelength of 2π to $10000 * 2\pi$.

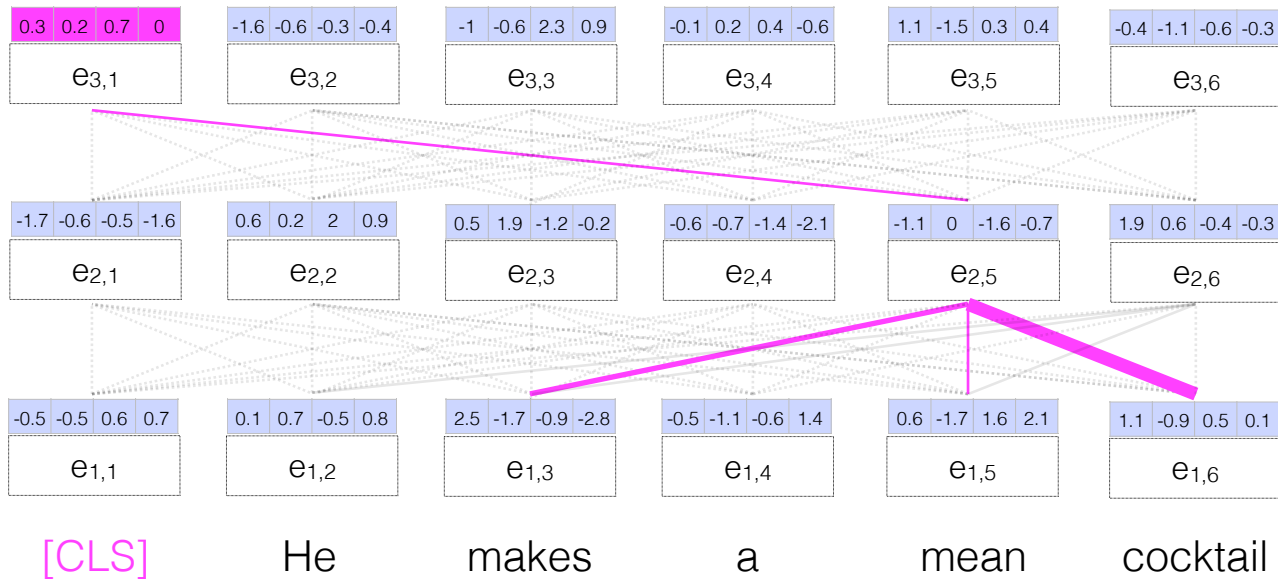
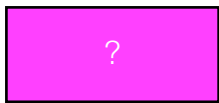
i = embedding
dimension

pos = position in
sequence

$$PE_{(pos, 2i)} = \sin(pos/10000^{2i/d_{\text{model}}})$$

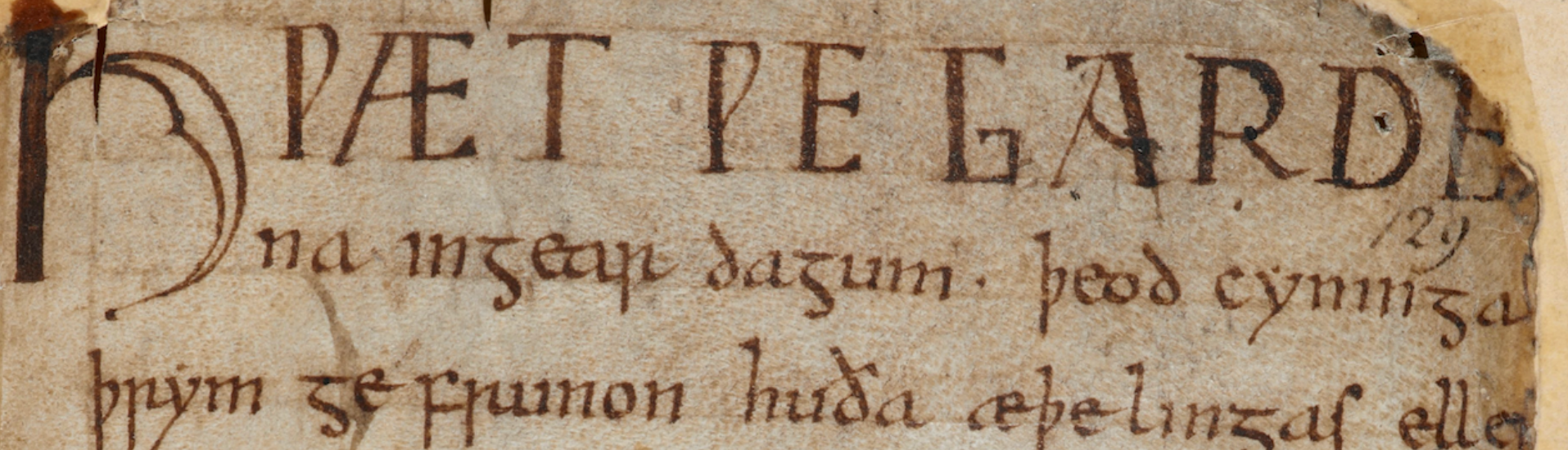
$$PE_{(pos, 2i+1)} = \cos(pos/10000^{2i/d_{\text{model}}})$$





Transformers

- Transformers have been extremely influential in NLP (Vaswani et al. 2017 has 35K citations!)
- We'll see them much more in this class in the context of specific applications:
 - Contextual language models, including causal self-attention (GPT), and bidirectional attention (BERT).
 - Machine translation
 - Text generation



Natural Language Processing

Info 159/259

Lecture 6: Annotation (Feb 3, 2022)

David Bamman, UC Berkeley

Modern NLP is driven by annotated data

- **Penn Treebank** (1993; 1995;1999); morphosyntactic annotations of WSJ
- **OntoNotes** (2007–2013); syntax, predicate-argument structure, word sense, coreference
- **FrameNet** (1998–): frame-semantic lexica/annotations
- **MPQA** (2005): opinion/sentiment
- **SQuAD** (2016): annotated questions + spans of answers in Wikipedia

Modern NLP is driven by annotated data

- In most cases, the data we have is the product of human judgments.
 - What's the correct part of speech tag?
 - Syntactic structure?
 - Sentiment?

Ambiguity

“One morning I shot
an elephant in my pajamas”



Animal Crackers

Dogmatism

Fast and Horvitz (2016), "Identifying Dogmatism in Social Media: Signals and Models"

Given a comment, imagine you hold a well-informed, different opinion from the commenter in question. We'd like you to tell us how likely that commenter would be to engage you in a constructive conversation about your disagreement, where you each are able to explore the other's beliefs. The options are:

(5): It's unlikely you'll be able to engage in any substantive conversation. When you respectfully express your disagreement, they are likely to ignore you or insult you or otherwise lower the level of discourse.

(4): They are deeply rooted in their opinion, but you are able to exchange your views without the conversation degenerating too much.

(3): It's not likely you'll be able to change their mind, but you're easily able to talk and understand each other's point of view.

(2): They may have a clear opinion about the subject, but would likely be open to discussing alternative viewpoints.

(1): They are not set in their opinion, and it's possible you might change their mind. If the comment does not convey an opinion of any kind, you may also select this option.

Sarcasm

“In many respects you know they honor President Obama. ISIS is honoring President Obama! He is the founder of ISIS. He’s the founder of ISIS, O.K.! He’s the founder, he founded ISIS and I would say the co-founder would be crooked Hillary Clinton. Co-founder, crooked Hillary Clinton. And that’s what it’s about.”



Donald J. Trump 
@realDonaldTrump

 Follow

Ratings challenged @CNN reports so seriously that I call President Obama (and Clinton) "the founder" of ISIS, & MVP. THEY DON'T GET SARCASM?

3:26 AM - Aug 12, 2016

 9,730  7,787  23,837

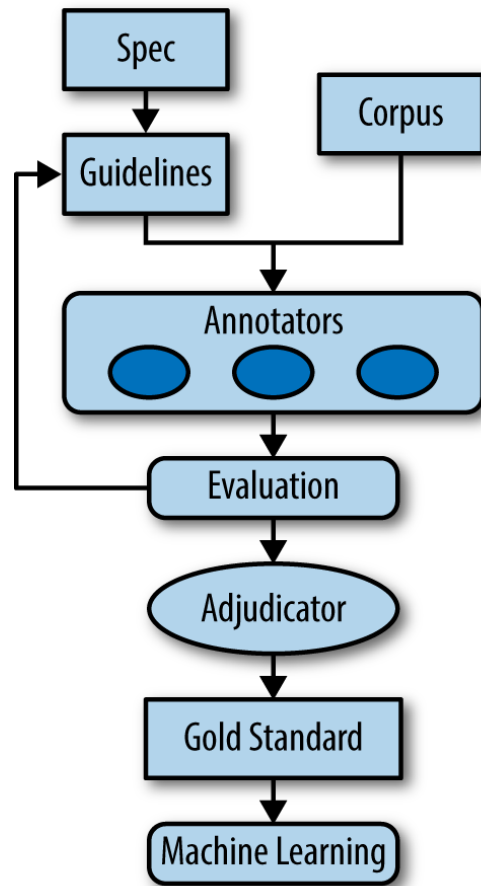


Fake News



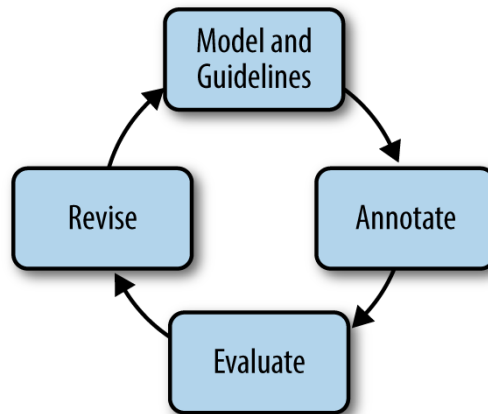
<http://www.fakenewschallenge.org>

Annotation pipeline



Pustejovsky and Stubbs (2012),
Natural Language Annotation for Machine Learning

Annotation pipeline



Pustejovsky and Stubbs (2012),
Natural Language Annotation for Machine Learning

Annotation guidelines

- Our goal: given the constraints of our problem, how can we formalize our description of the annotation process to encourage multiple annotators to provide the same judgment?

Annotation guidelines

- What is the goal of the project?
- What is each tag called and how is it used? (Be specific: provide examples, and discuss gray areas.)
- What parts of the text do you want annotated, and what should be left alone?
- How will the annotation be created? (For example, explain which tags or documents to annotate first, how to use the annotation tools, etc.)

Why not do it alone?

- Expensive/time-consuming
- Multiple people provide a measure of consistency: is the task well enough defined?
- Low agreement = not enough training, guidelines not well enough defined, task is bad

Adjudication

- Adjudication is the process of deciding on a single annotation for a piece of text, using information about the independent annotations.
- Can be as time-consuming (or more so) as a primary annotation.
- Does not need to be identical with a primary annotation (both annotators can be wrong by chance)

Interannotator agreement



annotator A

annotator B

	puppy	fried chicken
puppy	6	3
fried chicken	2	5

observed agreement = $11/16 = 68.75\%$

Cohen's kappa

- If classes are imbalanced, we can get high inter annotator agreement simply by chance

annotator A

	puppy	fried chicken
annotator B		
puppy	7	4
fried chicken	8	81

Cohen's kappa

- If classes are imbalanced, we can get high inter annotator agreement simply by chance

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

$$\kappa = \frac{0.88 - p_e}{1 - p_e}$$

annotator A

	puppy	fried chicken
annotator B		
puppy	7	4
fried chicken	8	81

Cohen's kappa

- Expected probability of agreement is how often we would expect two annotators to agree assuming **independent** annotations

$$\begin{aligned} p_e &= P(A = \text{puppy}, B = \text{puppy}) + P(A = \text{chicken}, B = \text{chicken}) \\ &= P(A = \text{puppy})P(B = \text{puppy}) + P(A = \text{chicken})P(B = \text{chicken}) \end{aligned}$$

Cohen's kappa

$$= P(A = \text{puppy})P(B = \text{puppy}) + P(A = \text{chicken})P(B = \text{chicken})$$

$$P(A=\text{puppy}) \quad 15/100 = 0.15$$

$$P(B=\text{puppy}) \quad 11/100 = 0.11$$

$$P(A=\text{chicken}) \quad 85/100 = 0.85$$

$$P(B=\text{chicken}) \quad 89/100 = 0.89$$

$$= 0.15 \times 0.11 + 0.85 \times 0.89$$

$$= 0.773$$

annotator A

	puppy	fried chicken
annotator B puppy	7	4
fried chicken	8	81

Cohen's kappa

- If classes are imbalanced, we can get high inter annotator agreement simply by chance

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

$$\kappa = \frac{0.88 - p_e}{1 - p_e}$$

$$\kappa = \frac{0.88 - 0.773}{1 - 0.773}$$

$$= 0.471$$

annotator A

	puppy	fried chicken
annotator B		
puppy	7	4
fried chicken	8	81

Cohen's kappa

- “Good” values are subject to interpretation, but rule of thumb:

0.80-1.00	Very good agreement
0.60-0.80	Good agreement
0.40-0.60	Moderate agreement
0.20-0.40	Fair agreement
< 0.20	Poor agreement

annotator A

annotator B

	puppy	fried chicken
puppy	0	0
fried chicken	0	100

annotator A

annotator B

	puppy	fried chicken
puppy	50	0
fried chicken	0	50

annotator A

annotator B

	puppy	fried chicken
puppy	0	50
fried chicken	50	0

Interannotator agreement

- Cohen's kappa can be used for any number of classes.
- Still requires **two** annotators who evaluate the same items.
- Fleiss' kappa generalizes to **multiple** annotators, each of whom may evaluate **different** items (e.g., crowdsourcing)

Fleiss' kappa

- Same fundamental idea of measuring the observed agreement compared to the agreement we would expect by chance.
- With $N > 2$, we calculate agreement among **pairs** of annotators

$$\kappa = \frac{P_o - P_e}{1 - P_e}$$

Fleiss' kappa

Number of annotators who assign category j to item i

$$n_{ij}$$

For item i with n annotations, how many annotators agree, among all $n(n-1)$ possible pairs

$$P_i = \frac{1}{n(n-1)} \sum_{j=1}^K n_{ij}(n_{ij} - 1)$$

Fleiss' kappa

For item i with n annotations, how many annotators agree, among all $n(n-1)$ possible pairs

$$P_i = \frac{1}{n(n-1)} \sum_{j=1}^K n_{ij}(n_{ij} - 1)$$

Annotator				
A	B	C	D	
+	+	+	-	

agreeing pairs
of annotators →

A-B
B-A
A-C
C-A
B-C
C-B

Label	n_{ij}
+	3
-	1

$$P_i = \frac{1}{4(3)} (3(2) + 1(0))$$

Fleiss' kappa

Average agreement among all items

$$P_o = \frac{1}{N} \sum_{i=1}^N P_i$$

Probability of category j

$$p_j = \frac{1}{Nn} \sum_{i=1}^N n_{ij}$$

Expected agreement by chance — joint probability two raters pick the same label is the product of their independent probabilities of picking that label

$$P_e = \sum_{j=1}^K p_j^2$$

Krippendorff's alpha

- Kappa values still require categorical labels
- What about **real-valued** labels (e.g., Likert ratings, ordinal values)?

Krippendorff's alpha

	doc 1	doc 2	doc 3	doc 4
annotator A	5	5	5	1
annotator B	4	5	4	3

- We'll use the same principle that we used before: how much do our **observed** labels for a document differ from what we'd **expect** given the ratings we see?
- For real-valued ratings, we will also use distance metric to quantify how different two ratings are.

Observed

	doc 1	doc 2	doc 3	doc 4
annotator A	5	5	5	1
annotator B	4	5	4	3

... how often do we see another
with this label for that same item?

when one annotator
gives this label...

	1	3	4	5
1		1		
3	1			
4				2
5			2	2

Expected

	doc 1	doc 2	doc 3	doc 4
annotator A	5	5	5	1
annotator B	4	5	4	3

rating	count
1	1
3	1
4	2
5	4

Given this distribution of ratings overall, how often would we expect to see a pair of ratings together?

$$P(r_1 = 5) = 4/8$$

$$P(r_2 = 1) = 1/7$$

$$P(r_1 = 5, r_2 = 1) = 4/8 \times 1/7 = 1/14$$

normalize over 7 now instead of 8
because we already selected one

Expected

	doc 1	doc 2	doc 3	doc 4
annotator A	5	5	5	1
annotator B	4	5	4	3

rating	count
1	1
3	1
4	2
5	4

$$P(r_1 = 1, r_2 = 1) = ?$$

Expected

	doc 1	doc 2	doc 3	doc 4
annotator A	5	5	5	1
annotator B	4	5	4	3

... what's the probability of seeing another
with this label for that same item?

when one annotator
gives this label...

	1	3	4	5
1	0	1/56	1/28	1/14
3	1/56	0	1/28	1/14
4	1/28	1/28	1/28	1/7
5	1/14	1/14	1/7	3/14

Expected

	doc 1	doc 2	doc 3	doc 4
annotator A	5	5	5	1
annotator B	4	5	4	3

Transform these into expected counts by multiplying by the total number of annotations (8)

when one annotator gives this label...

... how often do we **expect** to see another with this label for that same item?

	1	3	4	5
1	0	1/7	2/7	4/7
3	1/7	0	2/7	4/7
4	2/7	2/7	2/7	8/7
5	4/7	4/7	8/7	12/7

Distance

	doc 1	doc 2	doc 3	doc 4
annotator A	5	5	5	1
annotator B	4	5	4	3

For real labels, we can use the squared distance as a measure of cost.

$$(r_1 - r_2)^2$$

	1	3	4	5
1	0	4	9	16
3	4	0	1	4
4	9	1	0	1
5	16	4	1	0

Krippendorff's alpha

$$1 - \frac{\text{sum} \left[\begin{array}{c} \text{observed} \\ \begin{bmatrix} & 1 & 3 & 4 & 5 \\ 1 & & 1 & & \\ 3 & 1 & & & \\ 4 & & & & 2 \\ 5 & & & 2 & 2 \end{bmatrix} \odot \begin{bmatrix} & 1 & 3 & 4 & 5 \\ 1 & 0 & 4 & 9 & 16 \\ 3 & 4 & 0 & 1 & 4 \\ 4 & 9 & 1 & 0 & 1 \\ 5 & 16 & 4 & 1 & 0 \end{bmatrix} \end{array} \right]}{\text{sum} \left[\begin{array}{c} \text{expected} \\ \begin{bmatrix} & 1 & 3 & 4 & 5 \\ 1 & 0 & 1/7 & 3/7 & 4/7 \\ 3 & 1/7 & 0 & 2/7 & 4/7 \\ 4 & 2/7 & 2/7 & 2/7 & 8/7 \\ 5 & 4/7 & 4/7 & 8/7 & 12/7 \end{bmatrix} \odot \begin{bmatrix} & 1 & 3 & 4 & 5 \\ 1 & 0 & 4 & 9 & 16 \\ 3 & 4 & 0 & 1 & 4 \\ 4 & 9 & 1 & 0 & 1 \\ 5 & 16 & 4 & 1 & 0 \end{bmatrix} \end{array} \right]}$$

$\odot$ = elementwise multiplication

Annotation Project

- Your annotation task must be one that requires **your human judgment** for the labels.
- Some NLP tasks don't require human judgment — **don't** consider these!
 - stock price prediction (will \$GOOG go up or down on 2/1/22?) can use historical stock prices as labels.
 - censorship prediction (will a post on a social media site be censored?) can train a model on posts paired with a label of whether or not that post was deleted by the platform.

Annotation Project

- You should manually be labeling your data, and not using algorithmic processes to do so.
- Let's say you're annotating all of the mentions of **commercial products** in text (e.g., "Tide", "BMW", "Nintendo Switch") and you have a list of 5000 products that you're looking for, so you write some code to read in that list and then automatically tag all mentions of those 5000 products in the data. This is not interesting! (There is no need to label data for this in order to train a supervised system; you could simply run your algorithm instead.)

Annotation Project

- Your labels should not be deterministic, but really require some human comprehension of the context.
- Let's say you're annotating how **suspenseful** a literary passage is, and every time a passage contains the words "thunder" you rate it a 3 for suspense; if it contains "gasped" you rate it 2; if it contains "anticipated" you rate it a 1; and 0 otherwise. If you can write an algorithm like this that can fully deterministically predict your gold labels correctly, then it's not interesting enough. (Again, there is no need to label data for this in order to train a supervised system; you could simply run your algorithm instead.)

Annotation Project

<https://bcourses.berkeley.edu/courses/1512978/pages/annotation-project-guidelines>