Natural Language Processing (NLP)

Deep Learning

Brad Quinton, Scott Chin

Aside: Categorical vs Binary Cross Entropy

- There was a question from last class about the use of Categorical Cross Entropy Loss (sometimes called SoftMax Loss) vs. Binary Cross Entropy

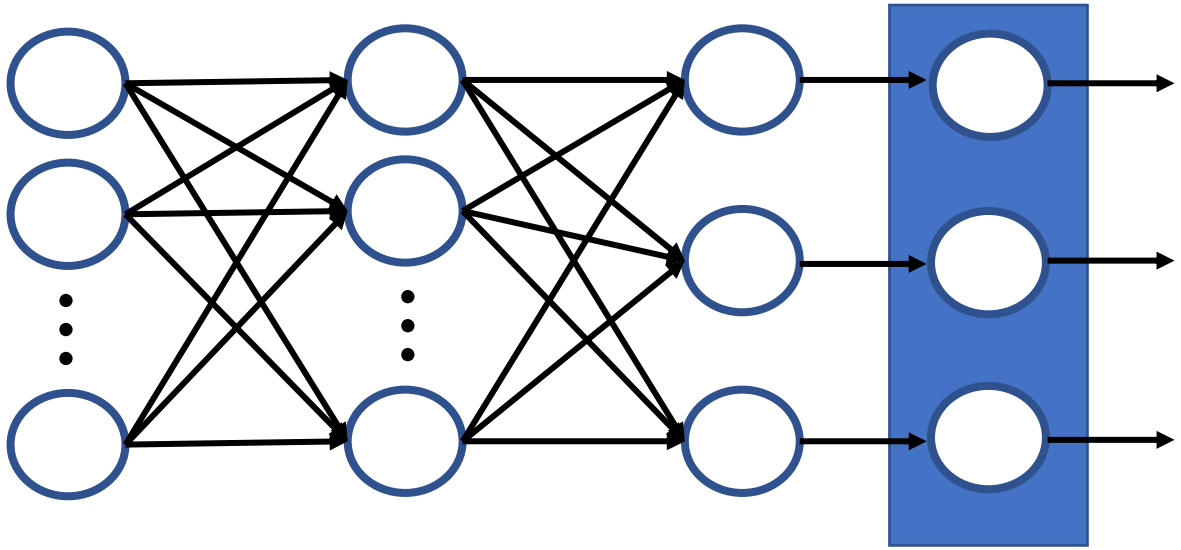
- This is bit subtle, but a common point of confusion that I see out there, so let's see if we can get some intuition…

- It is really a case of understanding what you want to do with your Neural Network outputs

Aside: Categorical vs Binary Cross Entropy

Sigmoid Output

$\sigma(z_i)$

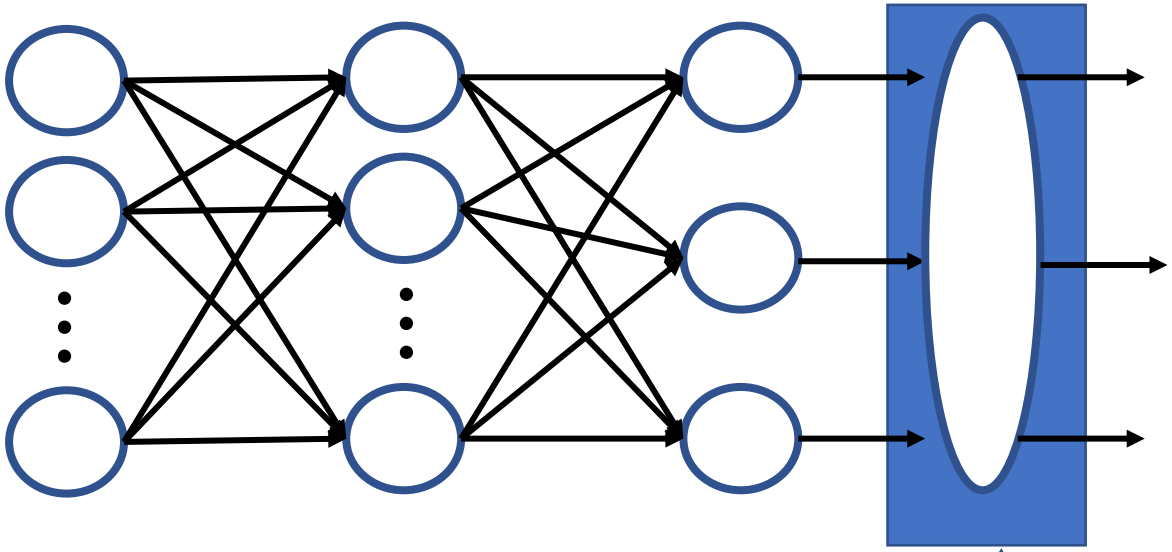


Inputs Hidden Layer Output Layer

Independent outputs, however the training labels can still be one-hot encoded.

Softmax

$$f_i(Z) = \frac{e^{z_i}}{\sum_{j=1}^{n_c} e^{z_j}}$$



Inputs Hidden Layer Output Layer

Dependent outputs, but the training labels must be one-hot encoded.

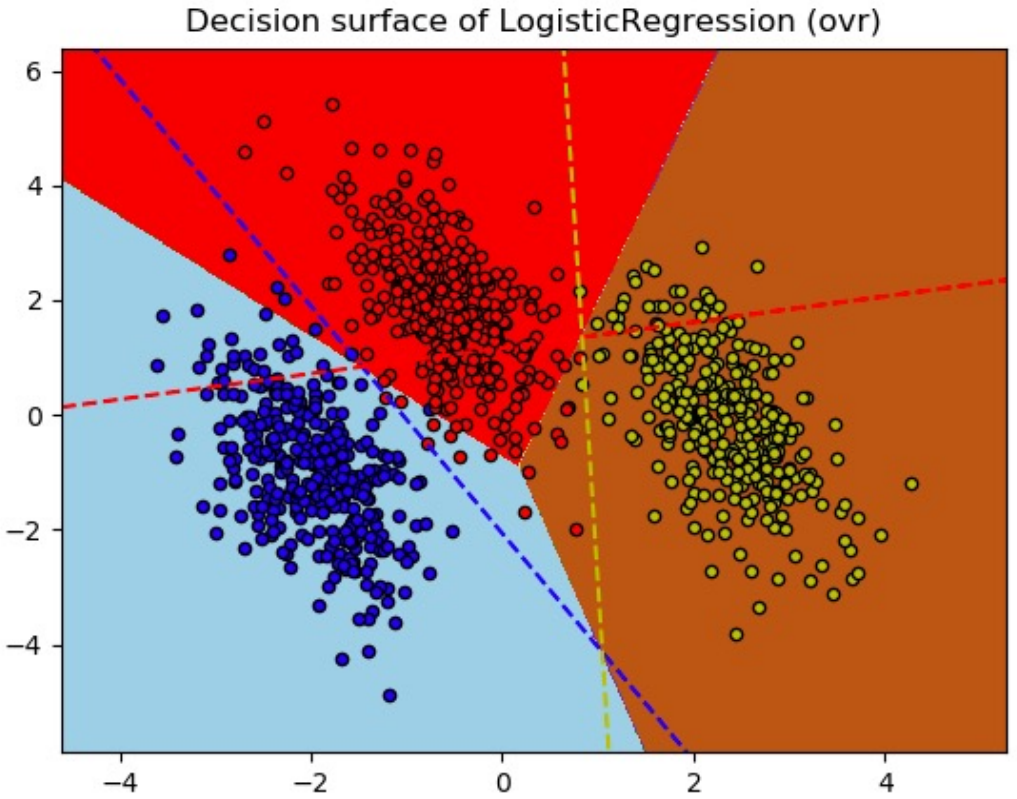
Aside: Categorical vs Binary Cross Entropy

SoftMax

- Classes are mutually exclusive

Sigmoid

- Classes may overlap, so that case must be interpreted
- For NLP overlap could equal UKW



Which is better? Depends on the goal of your learning system….

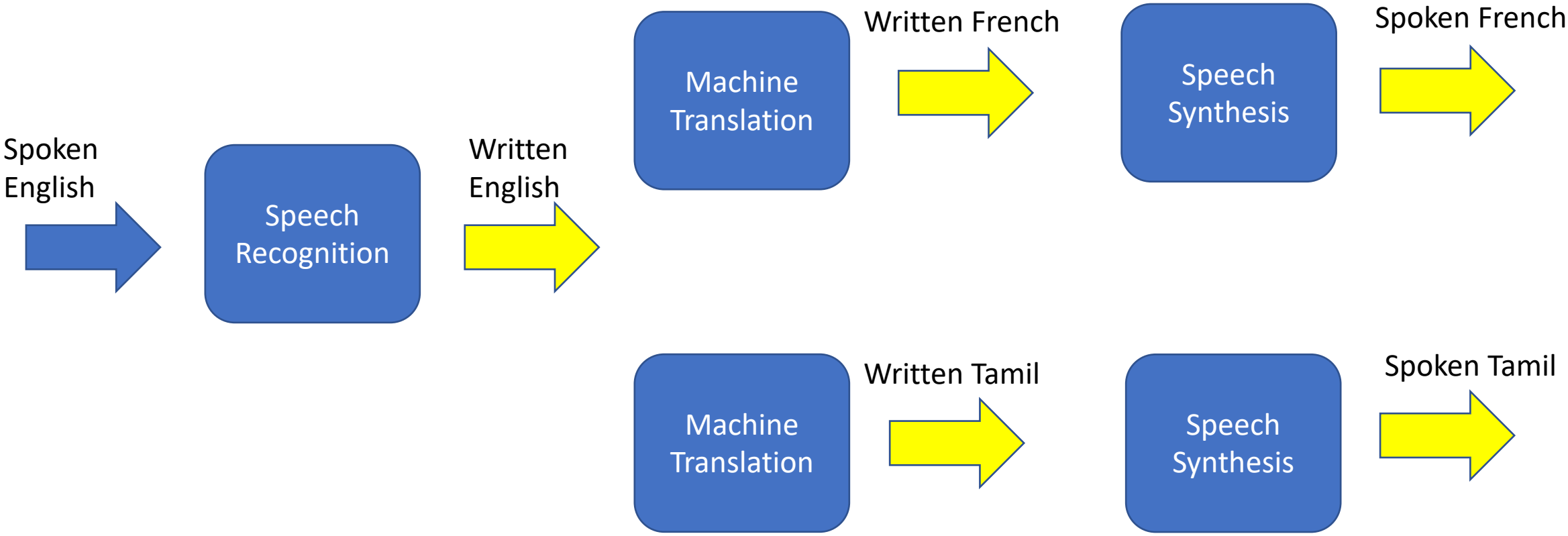
https://scikit-learn.org/stable/auto_examples/linear_model/plot_logistic_multinomial.html

Let’s remember how we got here…

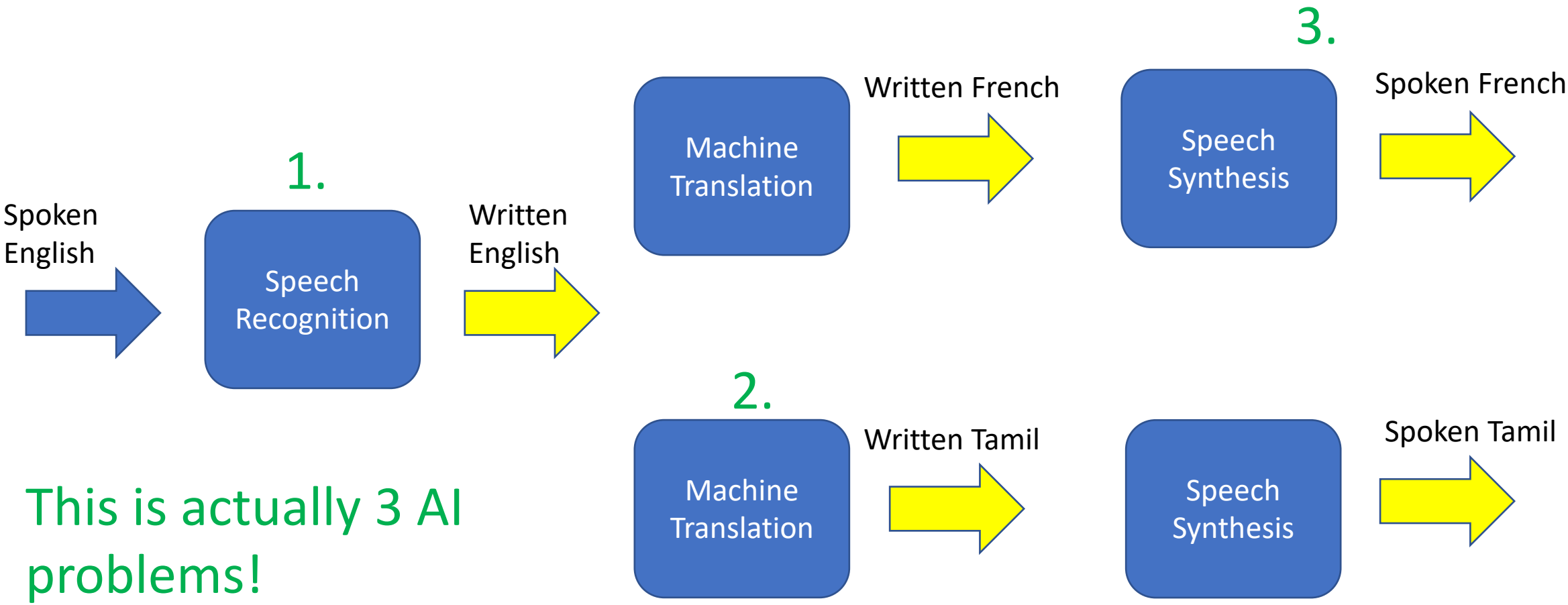
- We are trying to use AI to create a simultaneous, multi-language translator for basketball broadcasts as part of our fictious role as Director of AI for the Toronto Raptors

- We have been given this assignment by the CEO himself, and we really need to nail it to make a good impression!

Hypothetical case study….



Hypothetical case study….

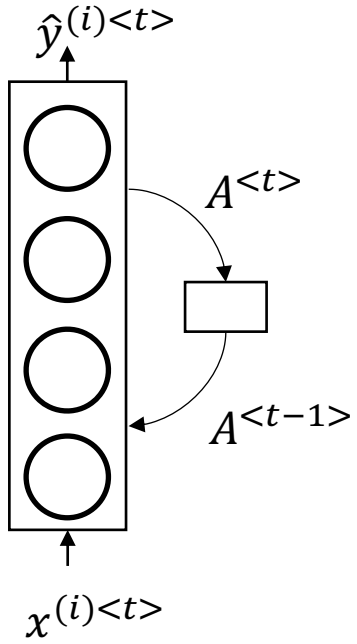


This is actually 3 AI problems!

Brad Quinton, Scott Chin

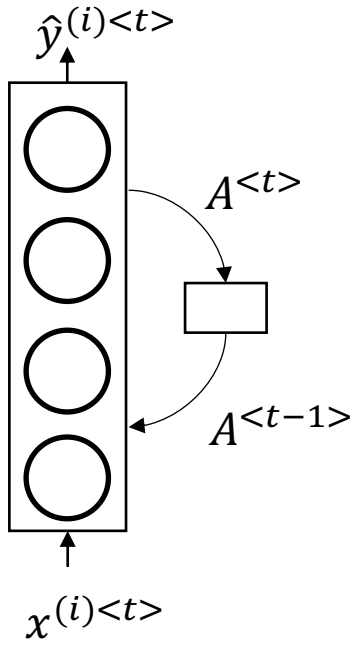
What We've Done So Far

- First, we realized that we needed to upgrade our Machine Learning architectures to interpret data like language that is naturally sequential, and context driven. This led us to the concept of RNNs



What We've Done So Far

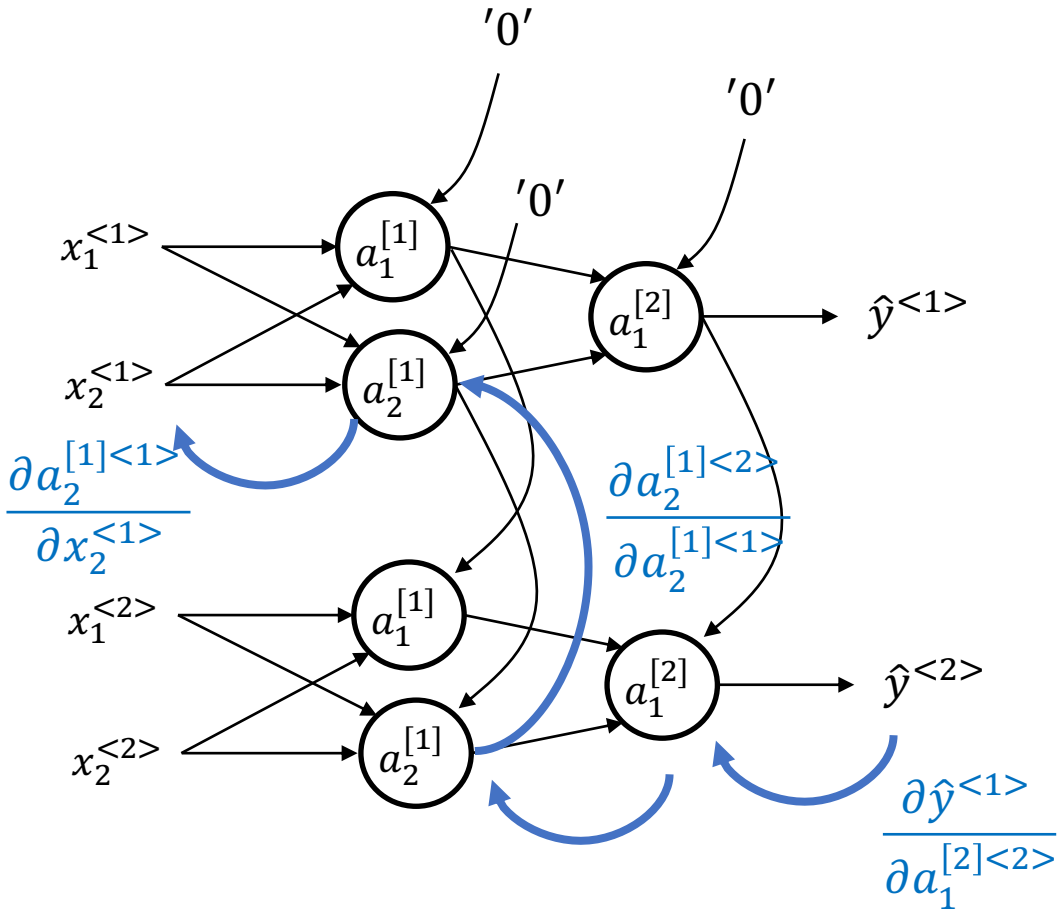
- First, we realized that we needed to upgrade our Machine Learning architectures to interpret data like language that is naturally sequential, and context driven. This led us to the concept of RNNs



Use the activations of the previous data in the sequence to bias the current interpretation.

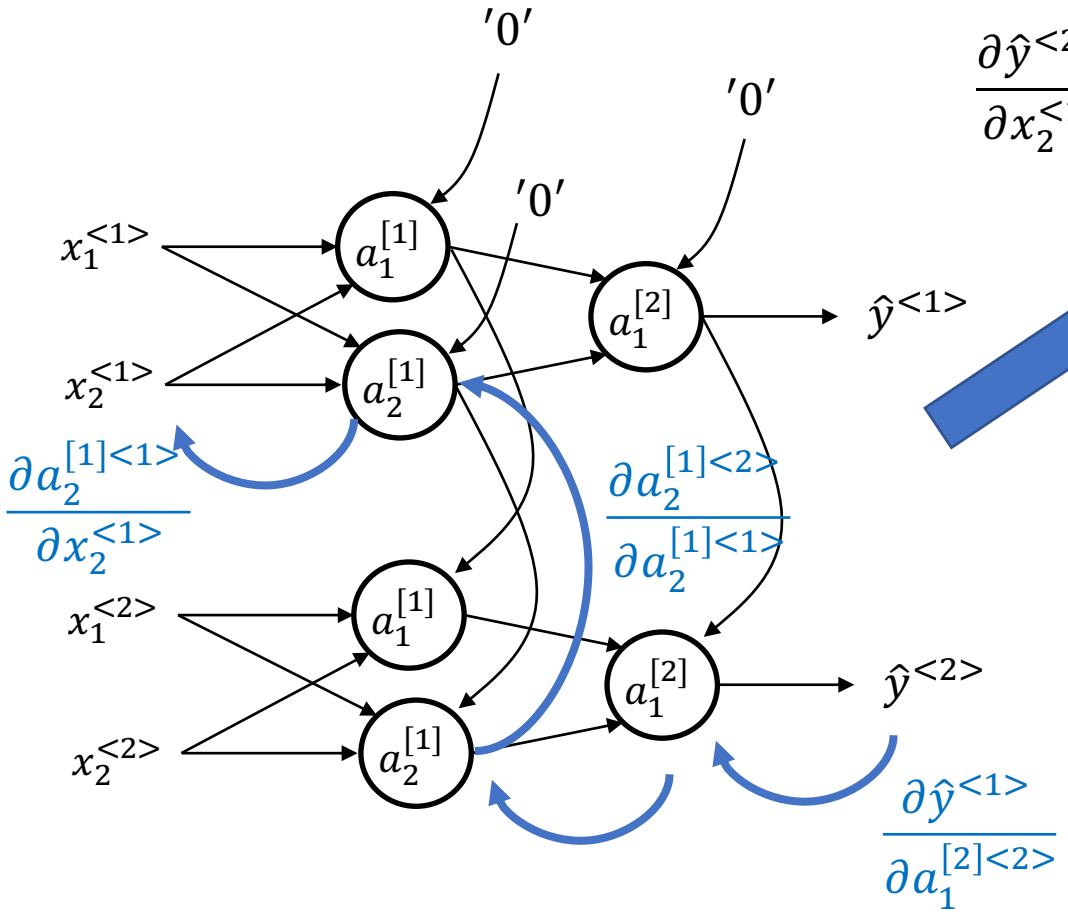
What We've Done So Far

- Then we worked through the details of implementing backprop on RNNs so we could train them.



What We've Done So Far

- Then we worked through the details of implementing backprop on RNNs so we could train them.

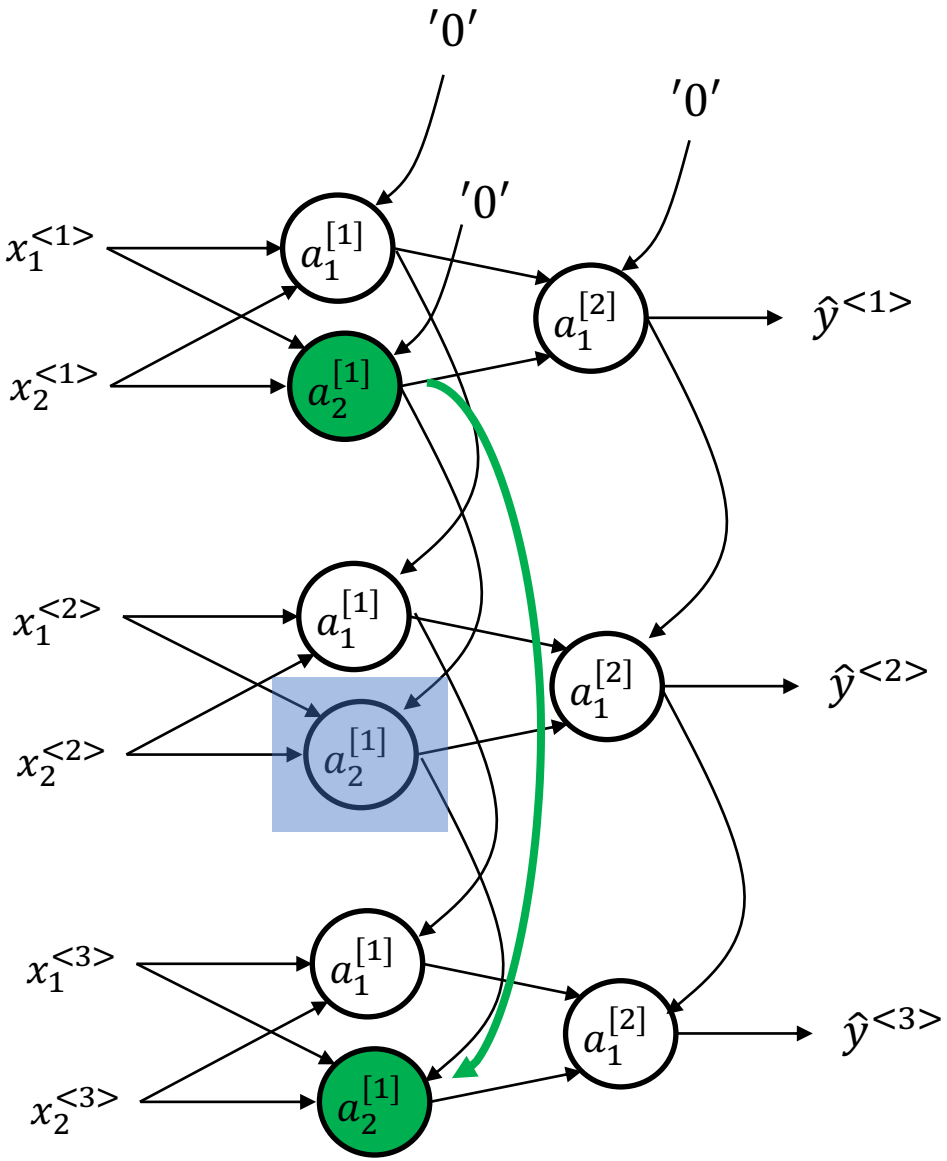


$$\frac{\partial \hat{y}^{<2>}}{\partial x_2^{<1>}} = \frac{\partial \hat{y}^{<2>}}{\partial a_1^{[2]<2>}} \cdot \frac{\partial a_1^{[2]<2>}}{\partial a_2^{[1]<2>}} \cdot \frac{\partial a_2^{[1]<2>}}{\partial a_2^{[1]<1>}} \cdot \frac{\partial a_2^{[1]<1>}}{\partial x_2^{<1>}}$$

By drawing out the computation graph we can use our previous work calculating gradients.

What We've Done So Far

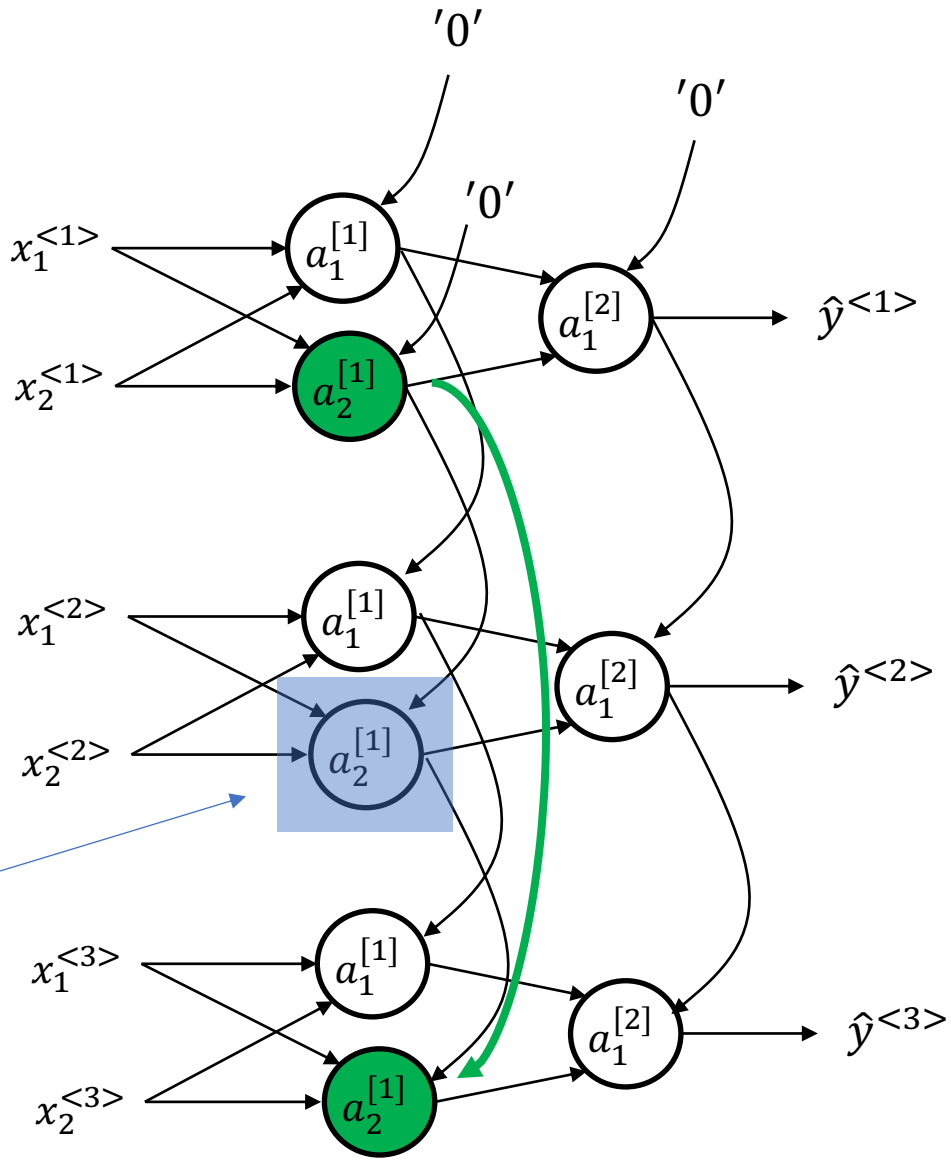
- Finally we've identified some critical issues in RNNs with vanishing gradients and added the GRU and LSTM architectures to address this



What We've Done So Far

- Finally we've identified some critical issues in RNNs with vanishing gradients and added the GRU and LSTM architectures to address this

Recall that we are repeated applying the *same* RNN activation units, so to "bypass" the current activation we simply hold the previous value. (*i.e.* we add a memory to our RNN)



Now What?

- The final gap we have for our hypothetical problem is to leverage all the Natural Language specific techniques that have been developed

- This is a BIG and quickly changing area (it could easily be a full course unto itself), so what I am going to do is to work through two key the ideas that I think are the most interesting and fundamental

- Hopefully, this will give you a good starting point if you decide to really dive into NLP

Word Encodings

- Recall our decision to use one-hot-encodings of words to avoid unintentional biases where some words were “closer” to others simply based on their order in the dictionary

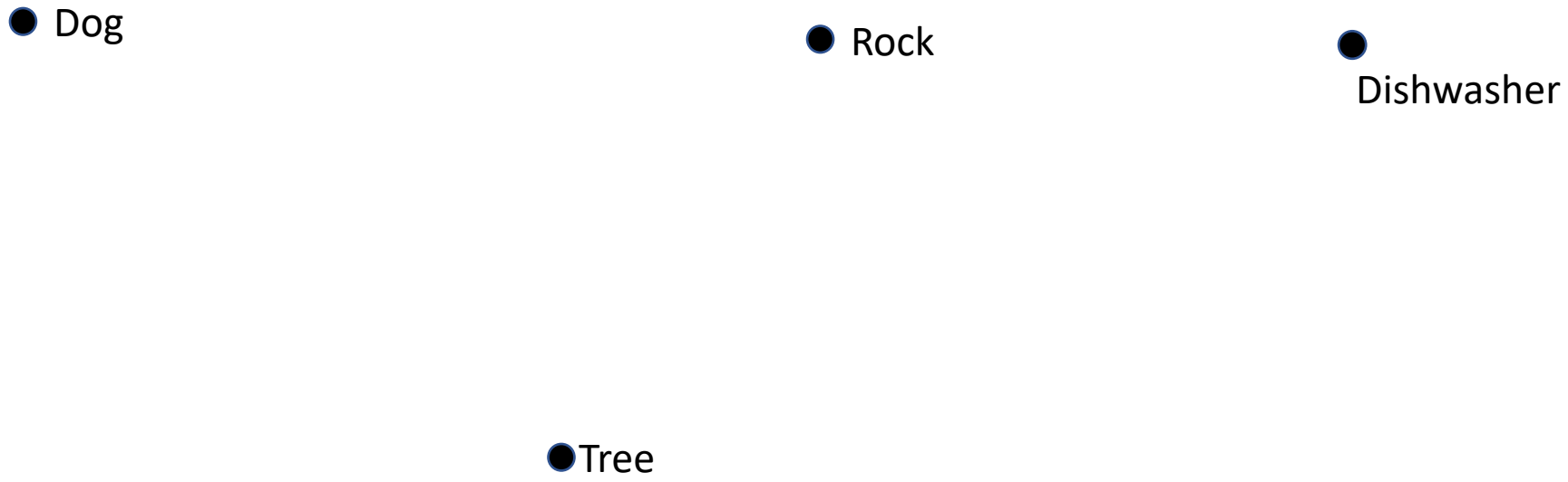
- Although it makes sense that words are not related (at least very much) based on their alphabetic order, it is not true to say words aren’t related… *some words really are “closer” to other words*

Word Encodings

- Let’s draw up some words and think about where we would place then on a hypothetical “closeness” map…

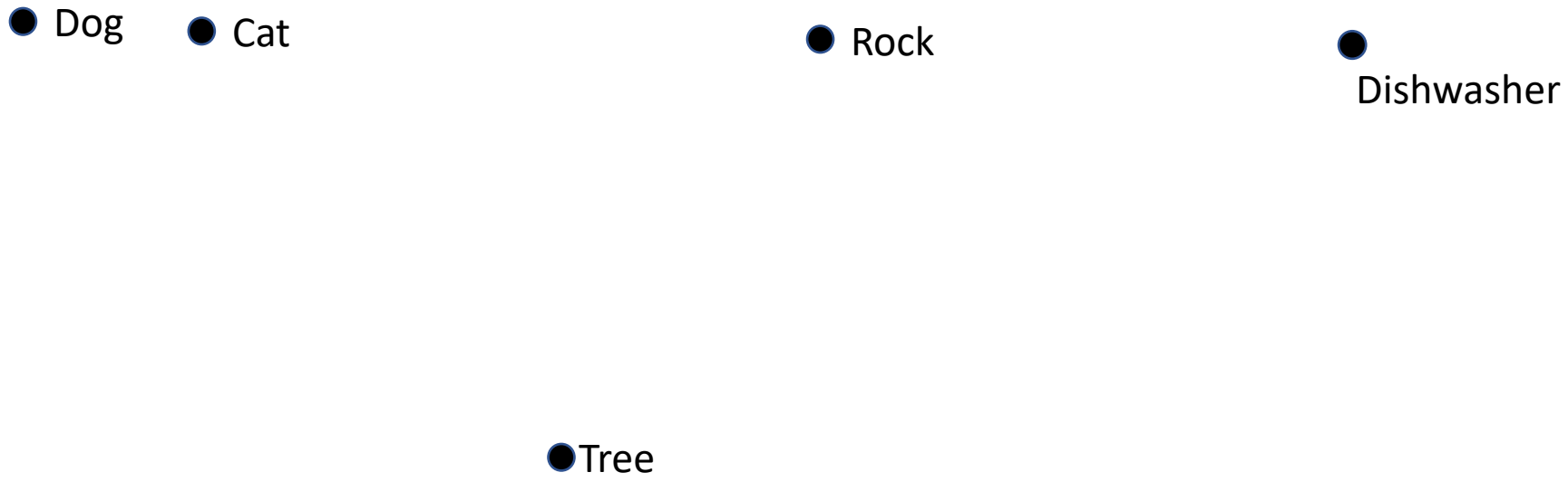
Word Encodings

- Let’s draw up some words and think about where we would place then on a hypothetical “closeness” map…



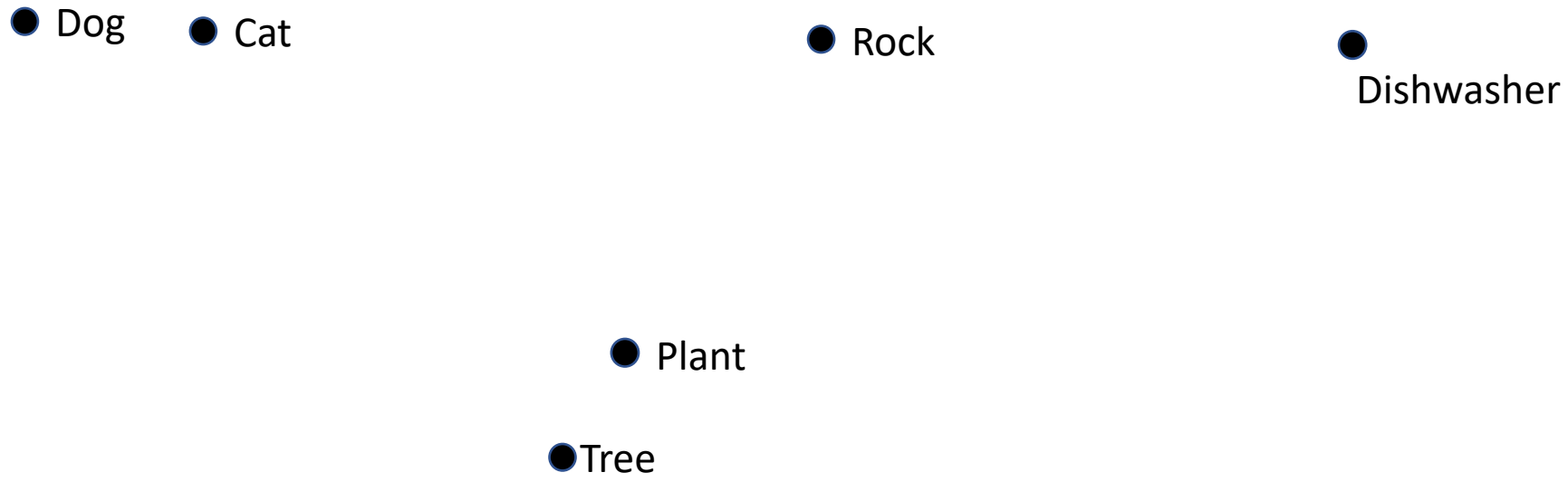
Word Encodings

- Let’s draw up some words and think about where we would place then on a hypothetical “closeness” map…



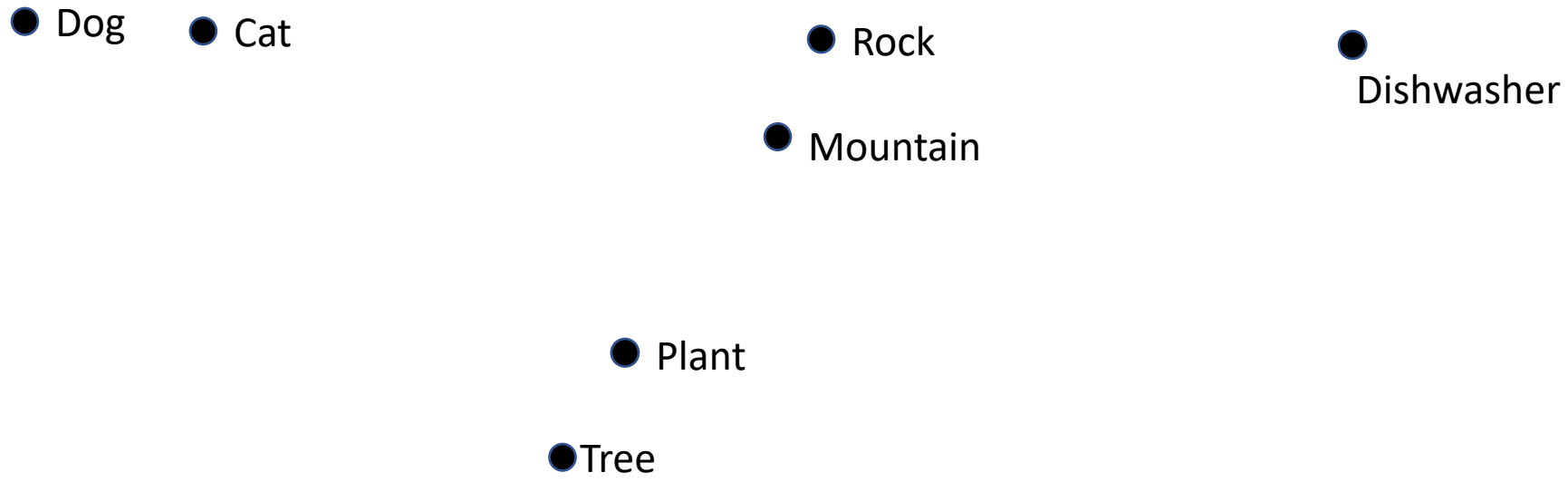
Word Encodings

- Let’s draw up some words and think about where we would place then on a hypothetical “closeness” map…



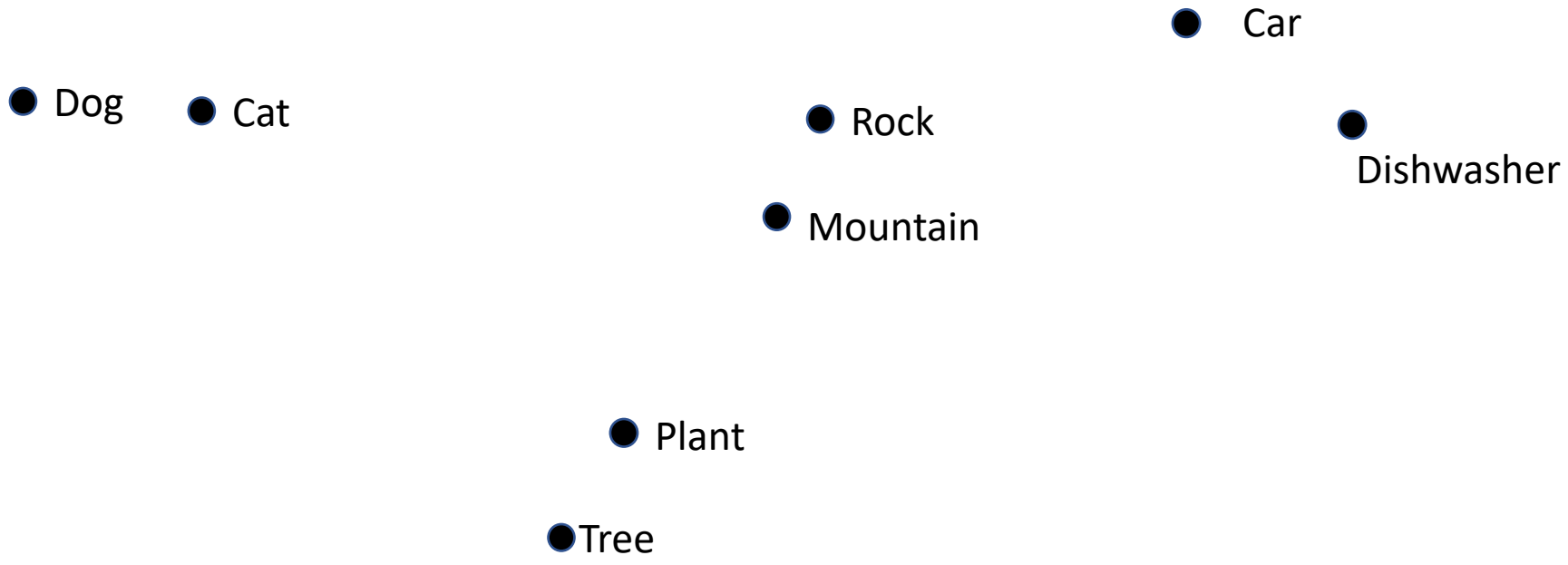
Word Encodings

- Let’s draw up some words and think about where we would place then on a hypothetical “closeness” map…



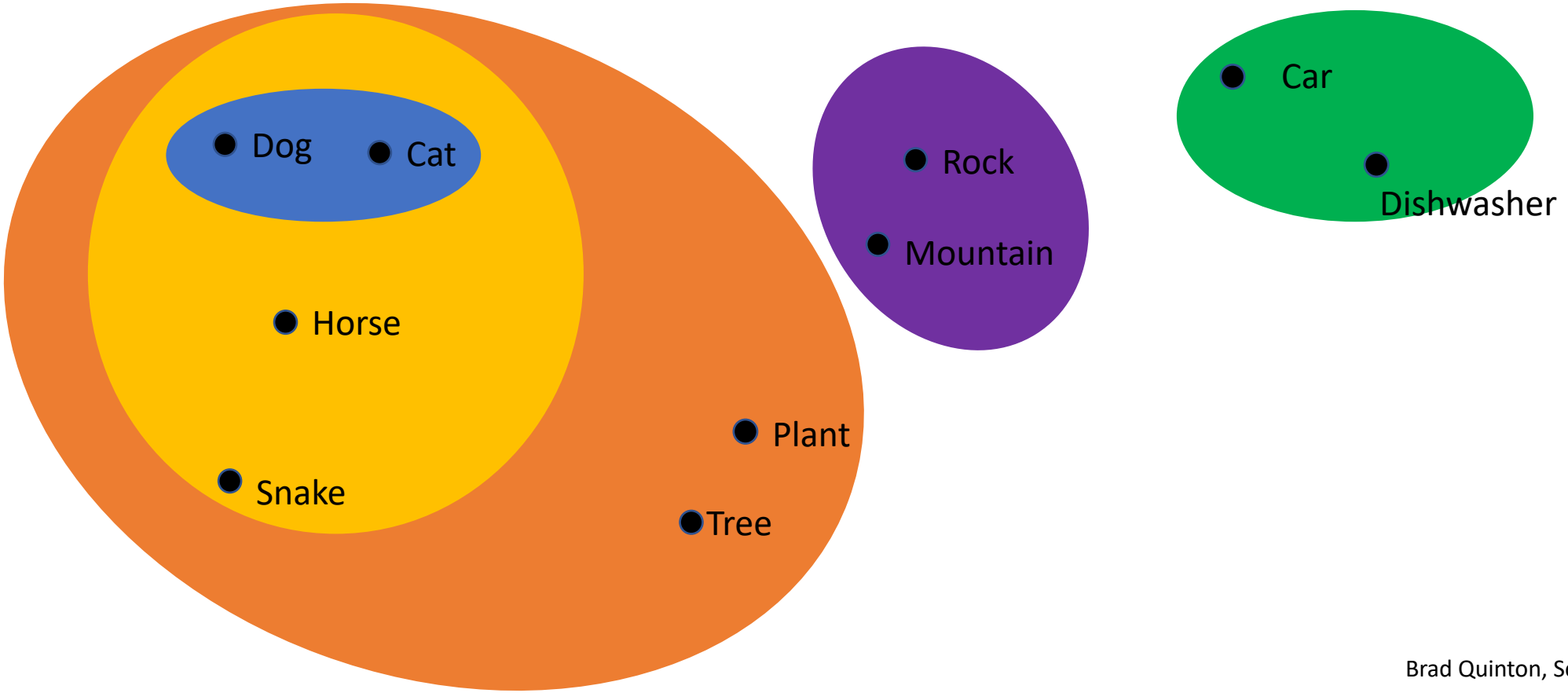
Word Encodings

- Let’s draw up some words and think about where we would place then on a hypothetical “closeness” map…



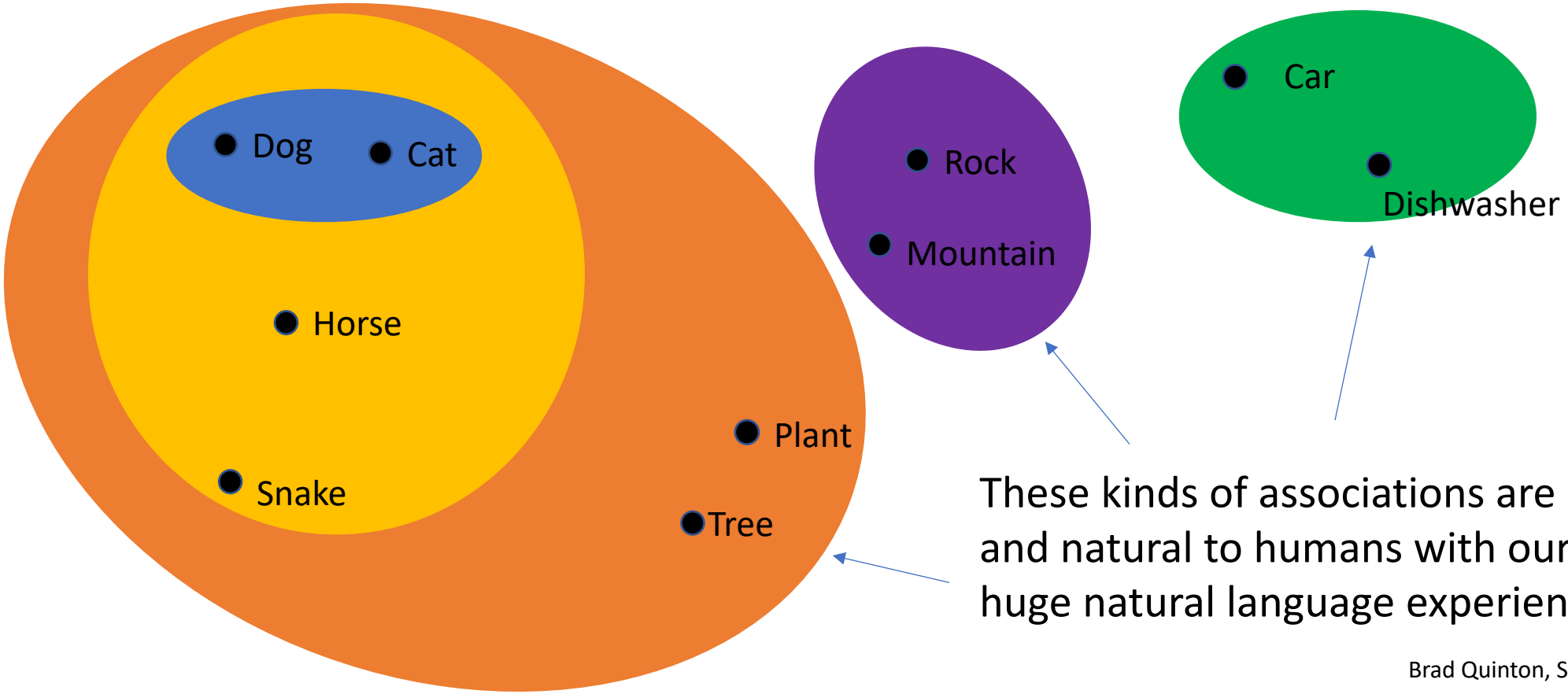
Word Encodings

- Let's draw up some words and think about where we would place then on a hypothetical "closeness" map…



Word Encodings

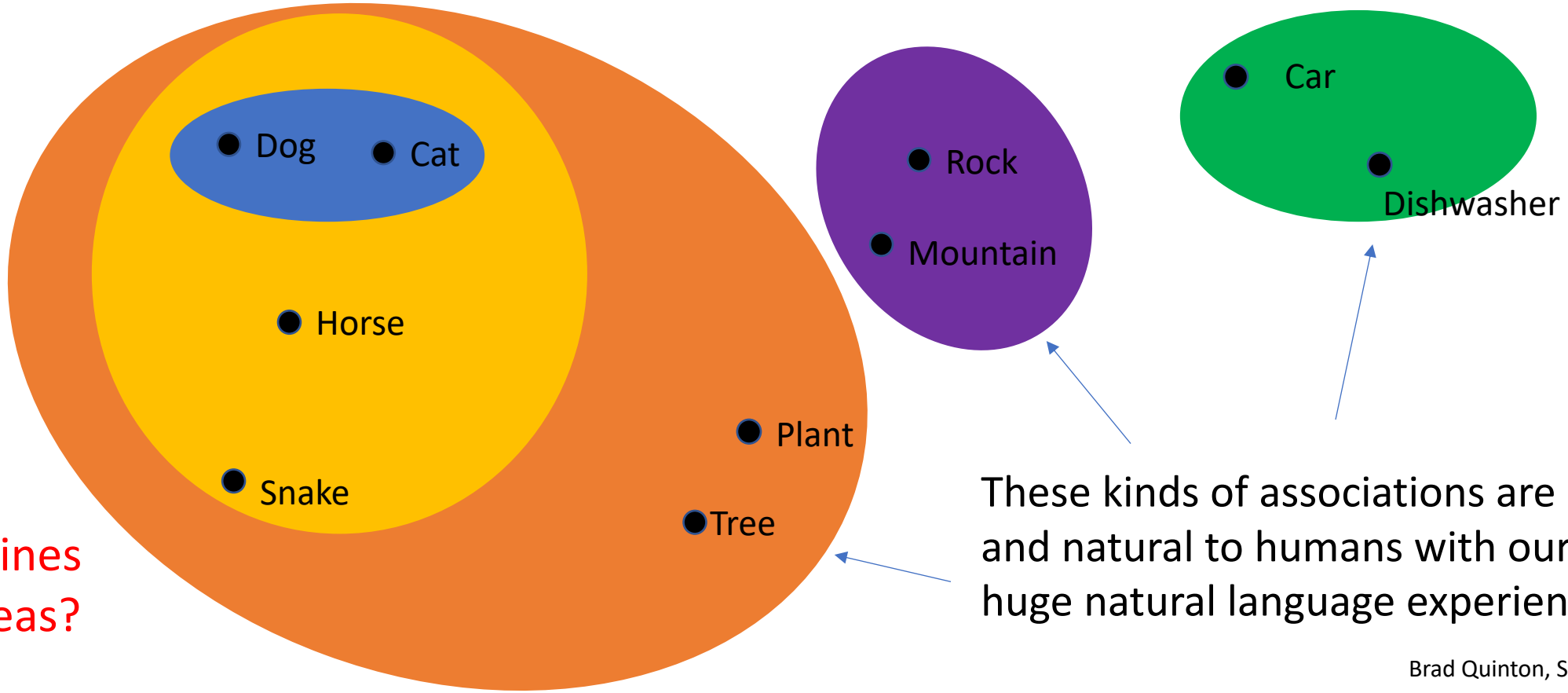
- Let's draw up some words and think about where we would place then on a hypothetical "closeness" map…



These kinds of associations are easy and natural to humans with our huge natural language experience.

Word Encodings

- Let's draw up some words and think about where we would place then on a hypothetical "closeness" map…



These kinds of associations are easy and natural to humans with our huge natural language experience.

But, can machines learn these ideas?

Word Encodings

- The somewhat surprising answer is: *yes!*

- Luckily, we have have created a huge amount of data reflecting our human understanding of word relationships in the corpus of written language that is available on-line

- Since we understand the meaning (and therefore the attributes) of words, the way we use them when we write reflects that understanding

Word Encodings

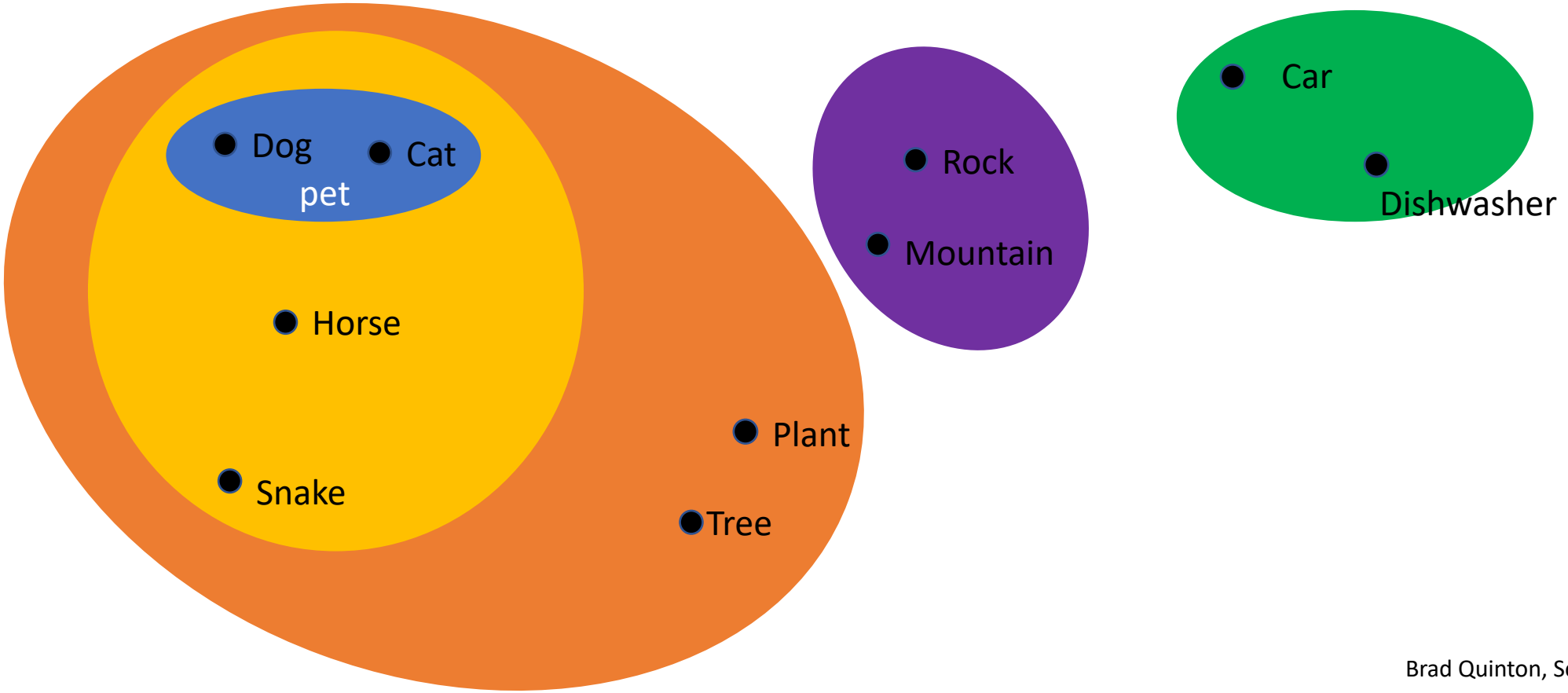
- So, where do we start?

- Rather than using a one-hot-encoding for each of the words in our vocabulary, we can imagine that for each word we have a vector where each element of the vector can be thought of as an attribute

- Let’s do a human-encoded example first….

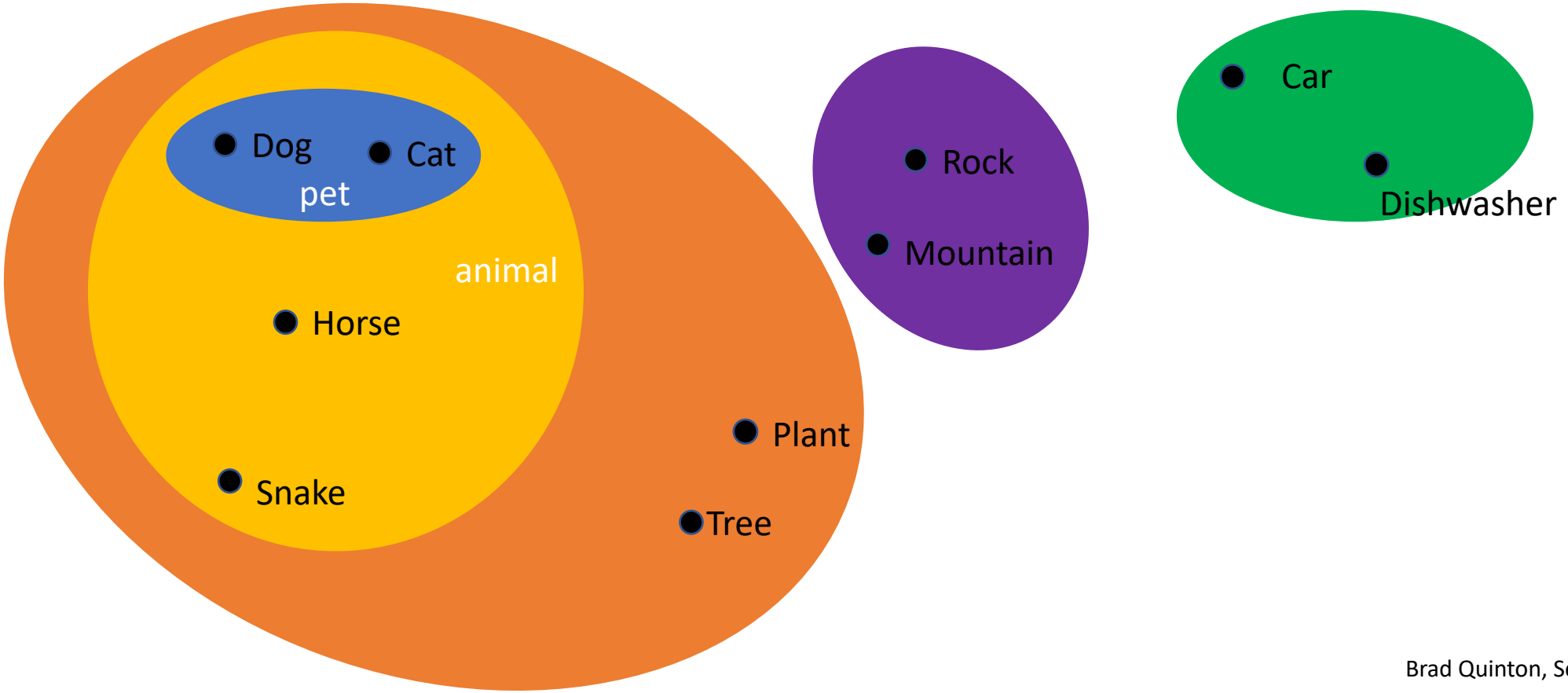
Word Encodings

- We can add some (arbitrary) labels to our groupings…



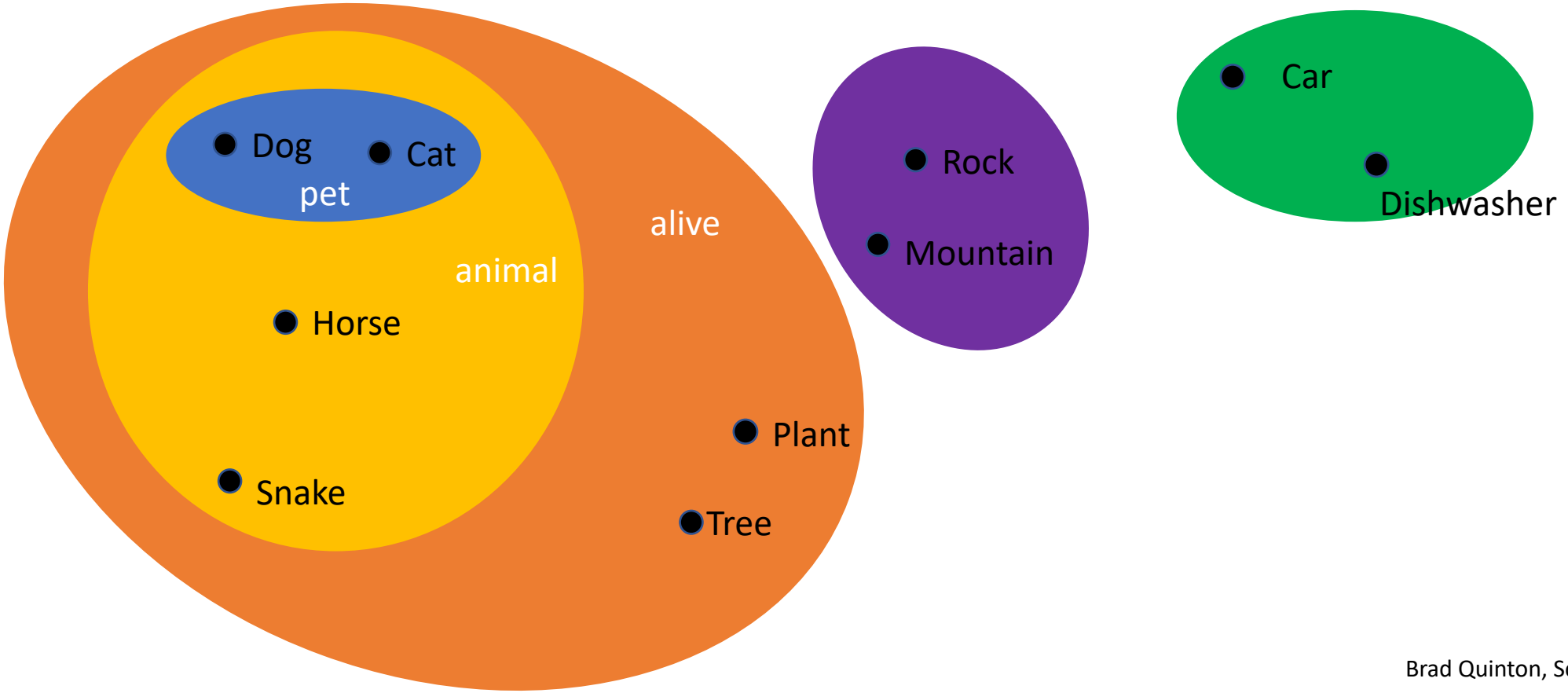
Word Encodings

- We can add some (arbitrary) labels to our groupings…



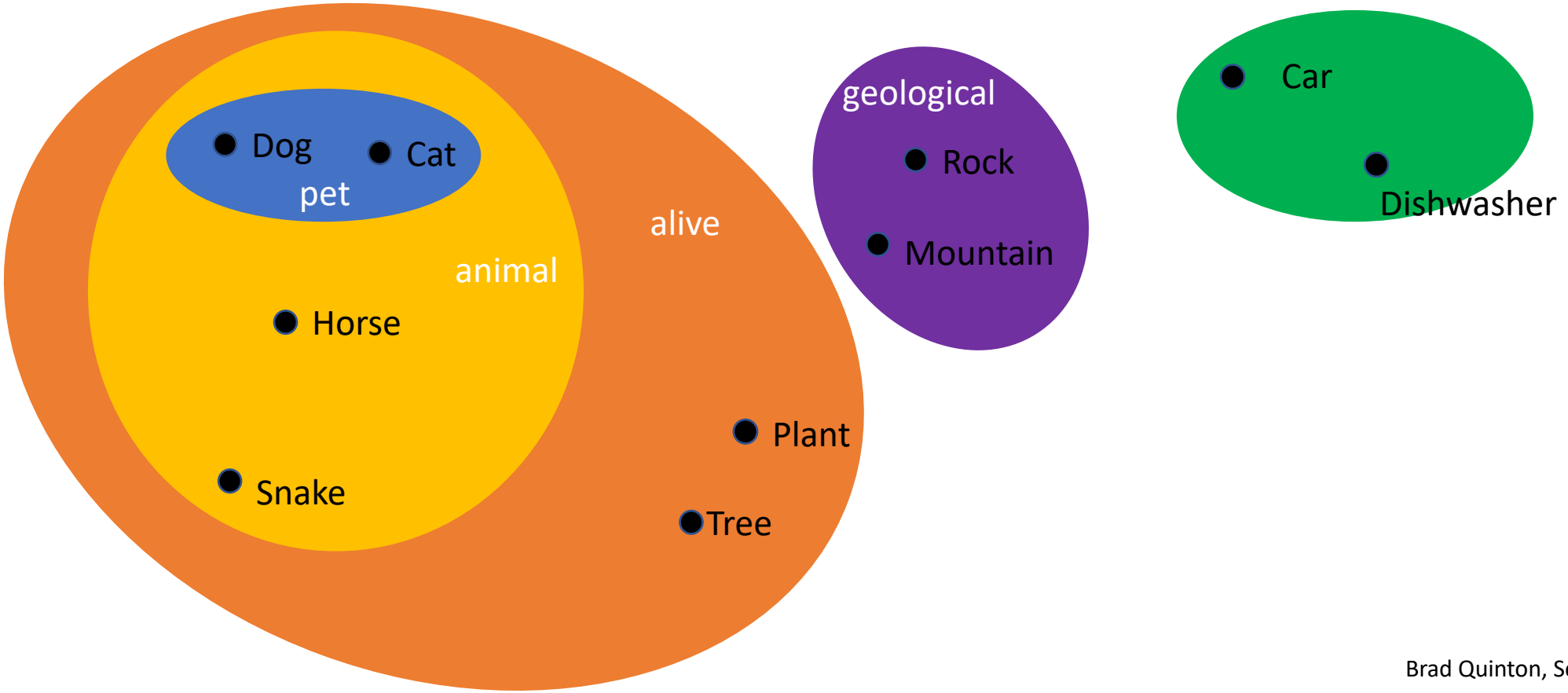
Word Encodings

- We can add some (arbitrary) labels to our groupings…



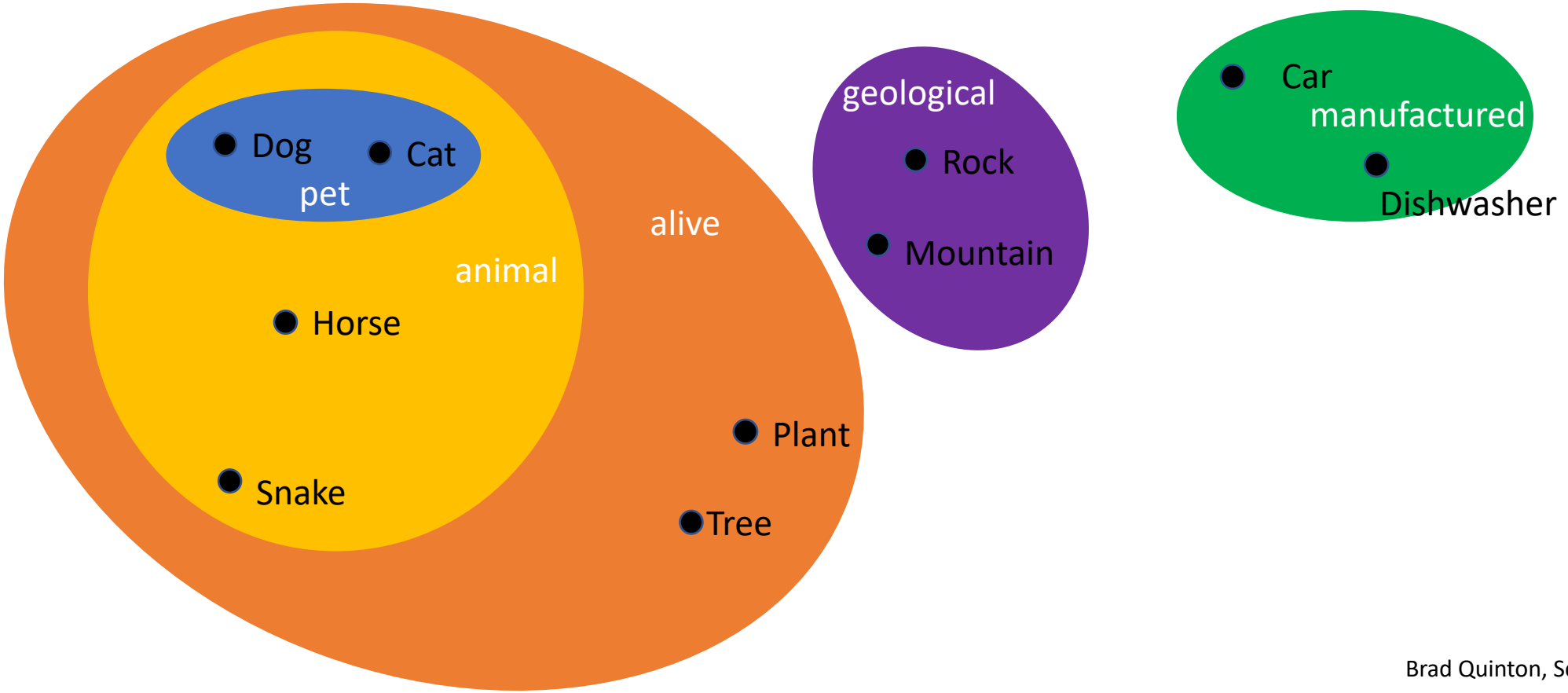
Word Encodings

- We can add some (arbitrary) labels to our groupings…



Word Encodings

- We can add some (arbitrary) labels to our groupings…



Word Encodings

	Pet	Animal	Alive	Geological	Manufactured
Dog	0.99				
Cat					
Horse					
Tree					
Plant					
Snake					
Rock					
Mountain					
Car					
Dishwasher					

Word Encodings

	Pet	Animal	Alive	Geological	Manufactured
Dog	0.99				
Cat	0.99				
Horse					
Tree					
Plant					
Snake					
Rock					
Mountain					
Car					
Dishwasher					

Word Encodings

	Pet	Animal	Alive	Geological	Manufactured
Dog	0.99				
Cat	0.99				
Horse	0.8				
Tree					
Plant					
Snake					
Rock					
Mountain					
Car					
Dishwasher					

Word Encodings

	Pet	Animal	Alive	Geological	Manufactured
Dog	0.99				
Cat	0.99				
Horse	0.8				
Tree	0.2				
Plant					
Snake					
Rock					
Mountain					
Car					
Dishwasher					

Word Encodings

	Pet	Animal	Alive	Geological	Manufactured
Dog	0.99	0.99	0.99	0.001	0.001
Cat	0.99	0.99	0.99	0.001	0.001
Horse	0.8	0.99	0.99	0.001	0.001
Tree	0.2	0.01	0.99	0.001	0.001
Plant	0.4	0.01	0.99	0.001	0.001
Snake	0.6	0.99	0.99	0.001	0.001
Rock	0.1	0.001	0.01	0.99	0.3
Mountain	0.01	0.001	0.05	0.99	0.01
Car	0.05	0.0001	0.01	0.001	0.99
Dishwasher	0.0001	0.0001	0.01	0.001	0.99

Word Encodings

	Pet	Animal	Alive	Geological	Manufactured
Dog	0.99	0.99	0.99	0.001	0.001
Cat	0.99	0.99	0.99	0.001	0.001
Horse	0.8	0.99	0.99	0.001	0.001
Tree	0.2	0.01	0.99	0.001	0.001
Plant	0.4	0.01	0.99	0.001	0.001
Snake	0.6	0.99	0.99	0.001	0.001
Rock	0.1	0.001	0.01	0.99	0.3
Mountain	0.01	0.001	0.05	0.99	0.01
Car	0.05	0.0001	0.01	0.001	0.99
Dishwasher	0.0001	0.0001	0.01	0.001	0.99

Word Encodings

	Pet	Animal	Alive	Geological	Manufactured
Dog	0.99	0.99	0.99	0.001	0.001
Cat	0.99	0.99	0.99	0.001	0.001
Horse	0.8	0.99	0.99	0.001	0.001
Tree	0.2	0.01	0.99	0.001	0.001
Plant	0.4	0.01	0.99	0.001	0.001
Snake	0.6	0.99	0.99	0.001	0.001
Rock	0.1	0.001	0.01	0.99	0.3
Mountain	0.01	0.001	0.05	0.99	0.01
Car	0.05	0.0001	0.01	0.001	0.99
Dishwasher	0.0001	0.0001	0.01	0.001	0.99

Word Encodings

We have built an implicit distance between Dogs and Dishwashers, but a closeness between Dog and Horses…

	Pet	Animal	Alive	Geological	Manufactured
Dog	0.99	0.99	0.99	0.001	0.001
Cat	0.99	0.99	0.99	0.001	0.001
Horse	0.8	0.99	0.99	0.001	0.001
Tree	0.2	0.01	0.99	0.001	0.001
Plant	0.4	0.01	0.99	0.001	0.001
Snake	0.6	0.99	0.99	0.001	0.001
Rock	0.1	0.001	0.01	0.99	0.3
Mountain	0.01	0.001	0.05	0.99	0.01
Car	0.05	0.0001	0.01	0.001	0.99
Dishwasher	0.0001	0.0001	0.01	0.001	0.99

Word Encodings

We want to **learn** the attribute groups

We have built an implicit distance between Dogs and Dishwashers, but a closeness between Dog and Horses…

	Pet	Animal	Alive	Geological	Manufactured
Dog	0.99	0.99	0.99	0.001	0.001
Cat	0.99	0.99	0.99	0.001	0.001
Horse	0.8	0.99	0.99	0.001	0.001
Tree	0.2	0.01	0.99	0.001	0.001
Plant	0.4	0.01	0.99	0.001	0.001
Snake	0.6	0.99	0.99	0.001	0.001
Rock	0.1	0.001	0.01	0.99	0.3
Mountain	0.01	0.001	0.05	0.99	0.01
Car	0.05	0.0001	0.01	0.001	0.99
Dishwasher	0.0001	0.0001	0.01	0.001	0.99

..and the vector values.

Word Encodings

We want to **learn** the attribute groups

We have built an implicit distance between Dogs and Dishwashers, but a closeness between Dog and Horses…

	Pet	Animal	Alive	Geological	Manufactured
Dog	0.99	0.99	0.99	0.001	0.001
Cat	0.99	0.99	0.99	0.001	0.001
Horse	0.8	0.99	0.99	0.001	0.001
Tree	0.2	0.01	0.99	0.001	0.001
Plant	0.4	0.01	0.99	0.001	0.001
Snake	0.6	0.99	0.99	0.001	0.001
Rock	0.1	0.001	0.01	0.99	0.3
Mountain	0.01	0.001	0.05	0.99	0.01
Car	0.05	0.0001	0.01	0.001	0.99
Dishwasher	0.0001	0.0001	0.01	0.001	0.99

..and the vector values.

This is called the Embedding Matrix, E

Brad Quinton, Scott Chin

Embedding Matrix

- We can pick the number of attributes that we think we will be sufficient to hold our encodings

- Let's assume assume, for instance, 250 attributes will work (this assumption is, of course, another hyperparameter that you could try tuning)

- Since our vocabulary is 20,000 words, then our embedding matrix, E will be of size (250, 20000)*

*Note using attributes for rows and words for columns this is the opposite of how I drew the human table, but it is more consistent with notation used by other research groups, so we will do it that way as well.

Learning Approach

- There are a number of different algorithms for learning the Embedding Matrix, E *:* Word2Vec, Negative Sampling, GloVe

- They all treat the elements of the matrix as parameters to be learned and use (not surprisingly) gradient descent to find a good solution

- There are many trade-offs between accuracy, run-time, etc. To get some intuition, let's look at a language model approach to learning E

Language Models

- Recall from previous discussions (for instance on GPT-2) that language models are used to predict language based on and previous inputs (*i.e.* the context)

- For example we might train a model to be able complete the sentence based on previous words such as:

The most popular type of pet is _____.

- We could, of course, use our previous one-hot-encodings to solve this problem

Language Models

- However, it is intuitive that an encoding that allowed, for instance, dogs and cats to be represented as similar would make the problem easier…

- First, notice that given how we have constructed the embedding matrix we can use the one-hot-encoding for each word to “extract” the vector for the specific word

Moving from One-hot to Encoding Vector

- If we name our one-hot-vectors as follows:

$$o_v$$

- where v is the position of the '1' in the one-hot vector, then

$$E \cdot o_v = e_v$$

- The e_v is the encoding of the v^{th} word in the vocabulary

Moving from One-hot to Encoding Vector

- If we name our one-hot-vectors as follows:

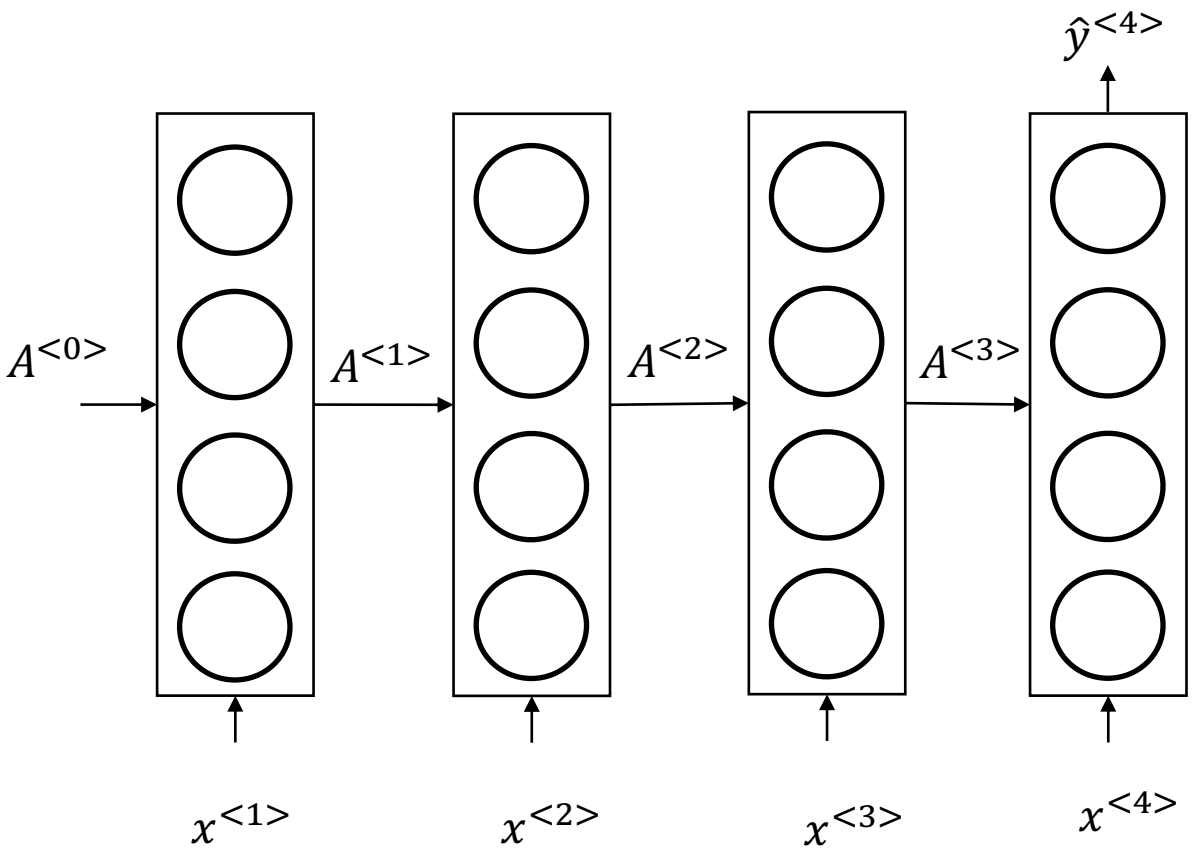
o_v (1,20000)

- where v is the position of the '1' in the one-hot vector, then

(250, 20000) $E \cdot o_v = e_v$ (250, 1)

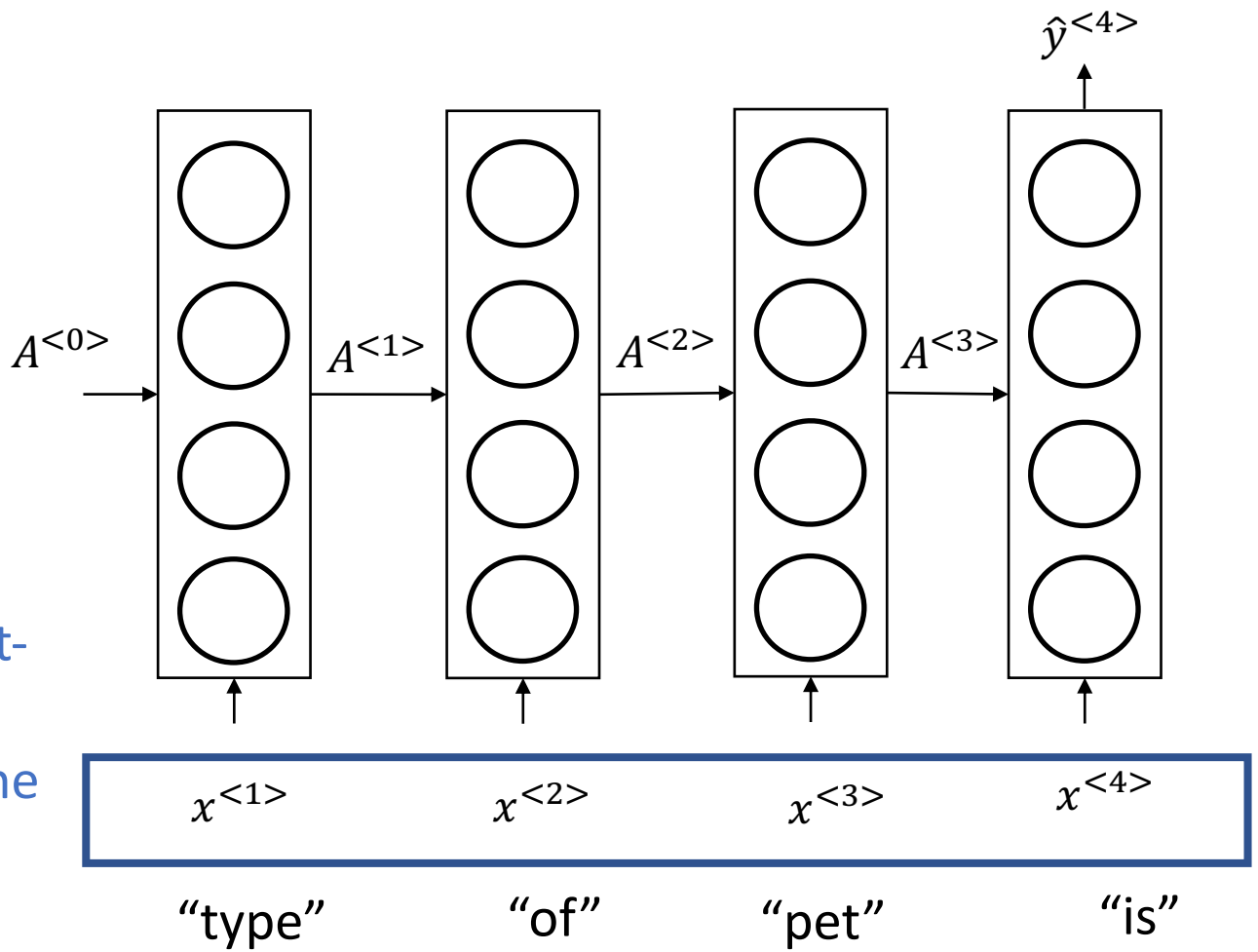
- The e_v is the encoding of the v^{th} word in the vocabulary

Language Models



Over a large set of training data (*i.e.* complete sentences) we would learn to predict the next word from the previous 4 words.

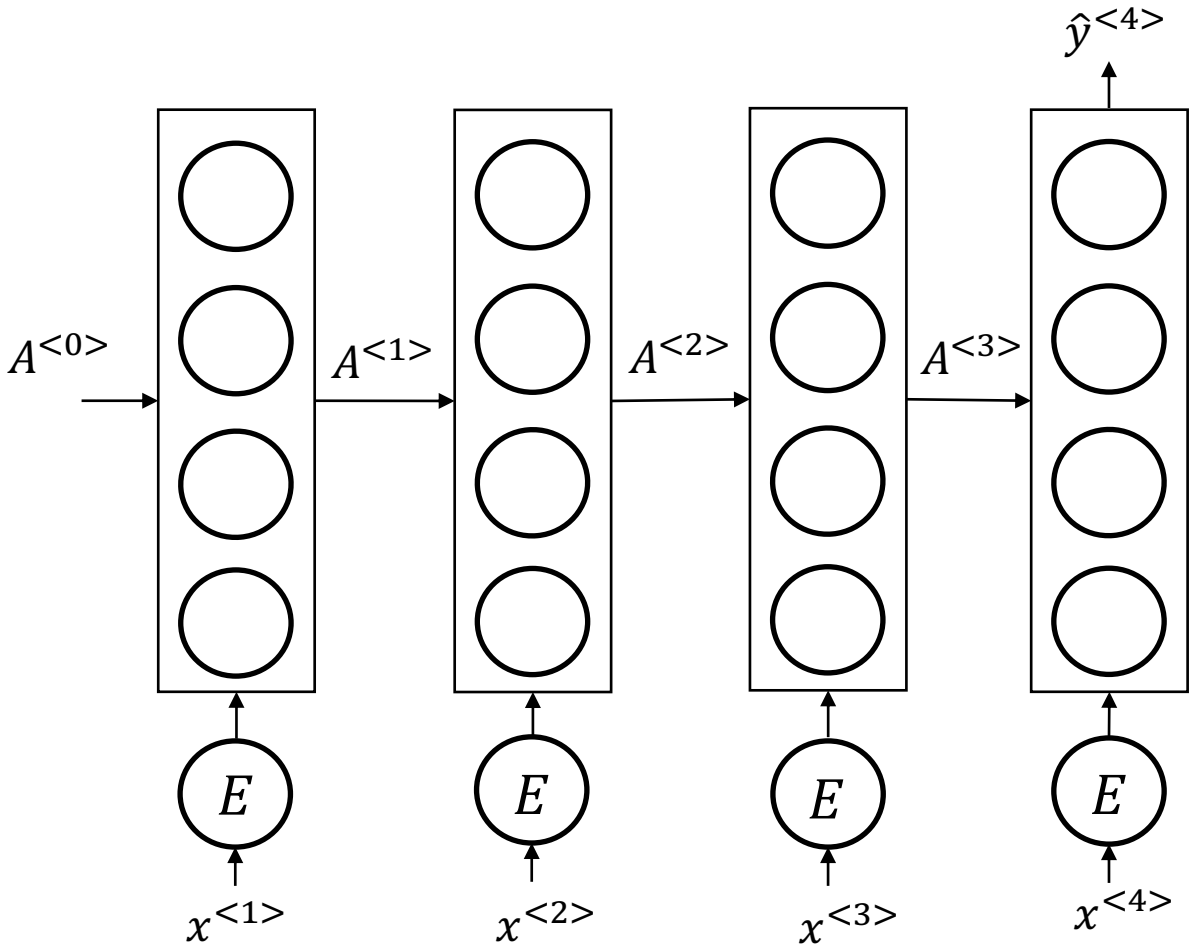
Language Models



Over a large set of training data (*i.e.* complete sentences) we would learn to predict the next word from the previous 4 words.

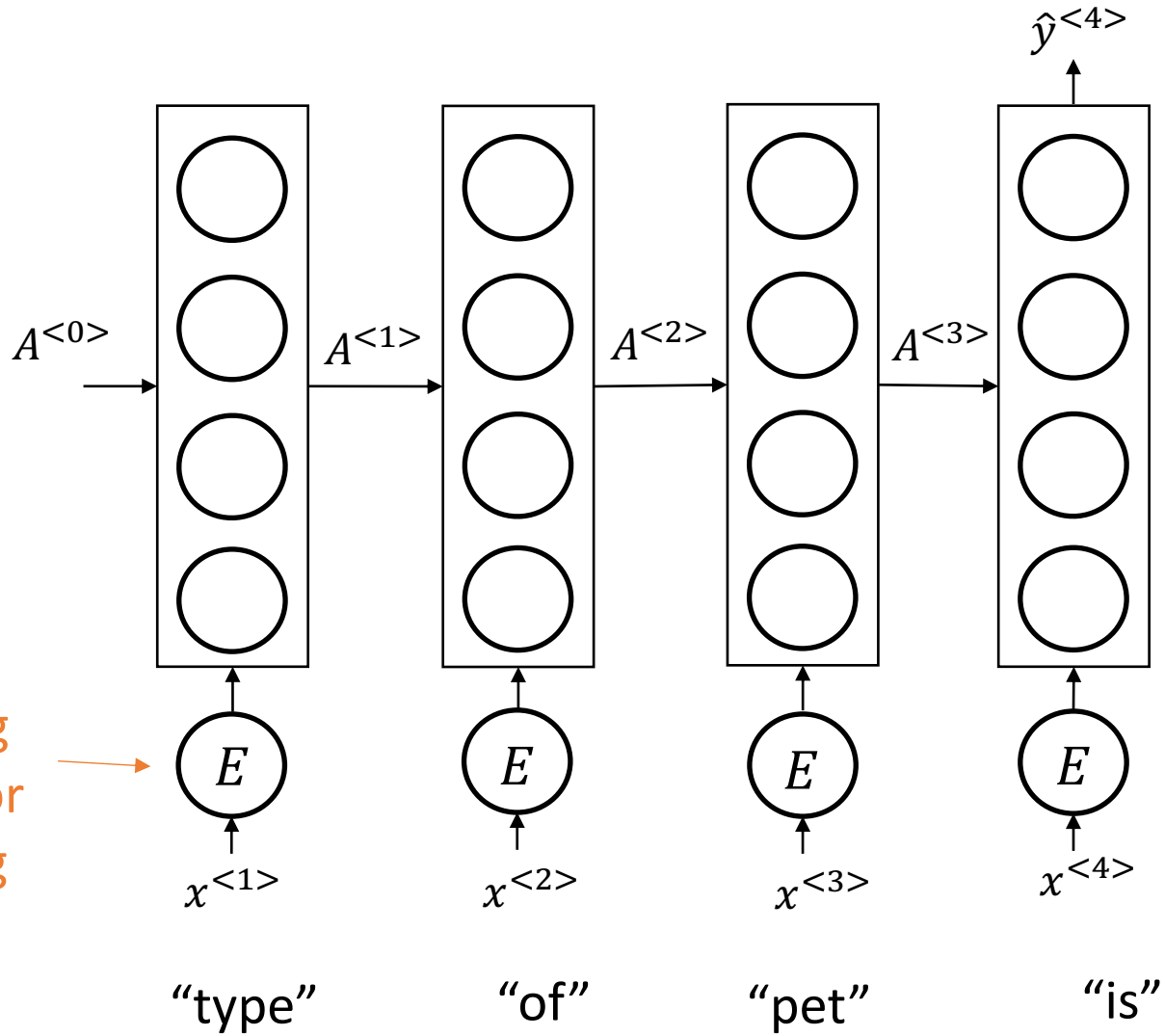
Normally these would be one-hot-encodings with length equal to the vocabulary size.

Adding Learnable Embedding Matrix

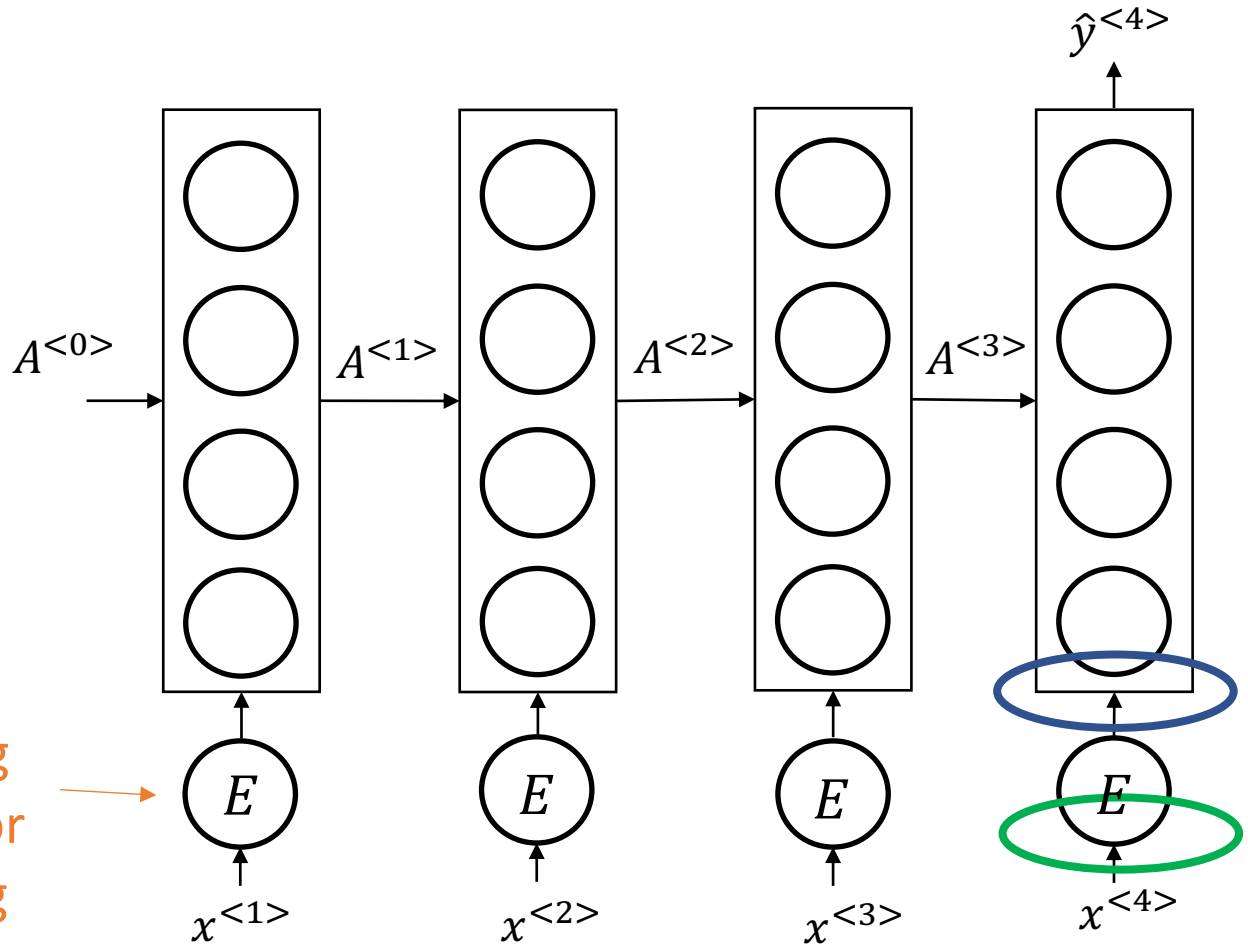


Adding Learnable Embedding Matrix

We can add an encoding step, by simply multiplying the one-hot-vector by the Embedding Matrix.



Adding Learnable Embedding Matrix

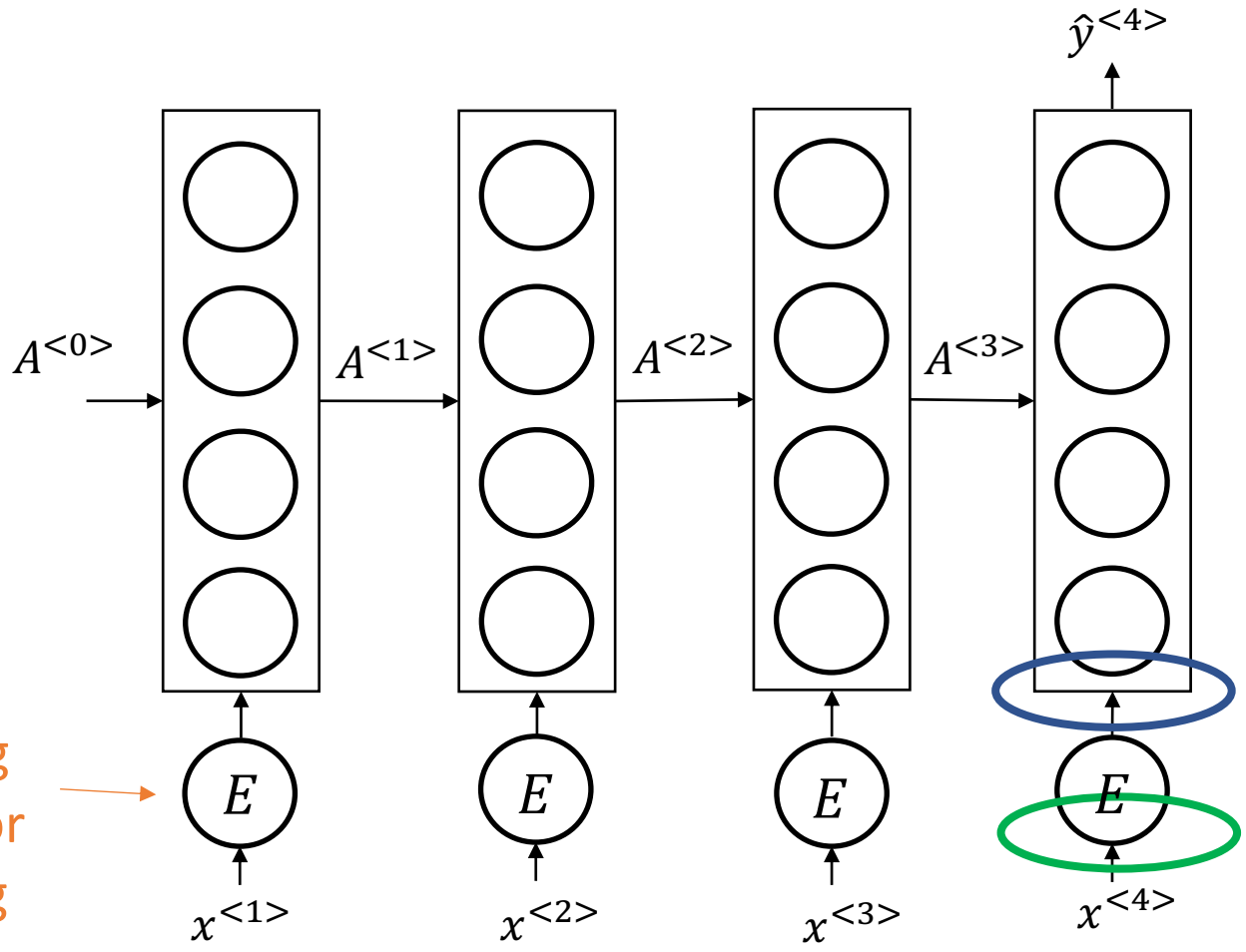


We can add an encoding step, by simply multiplying the one-hot-vector by the Embedding Matrix.

Vector size = height of Embedding Matrix (250)

Vector size = length of vocabulary (20,000)

Adding Learnable Embedding Matrix



Learned parameters:

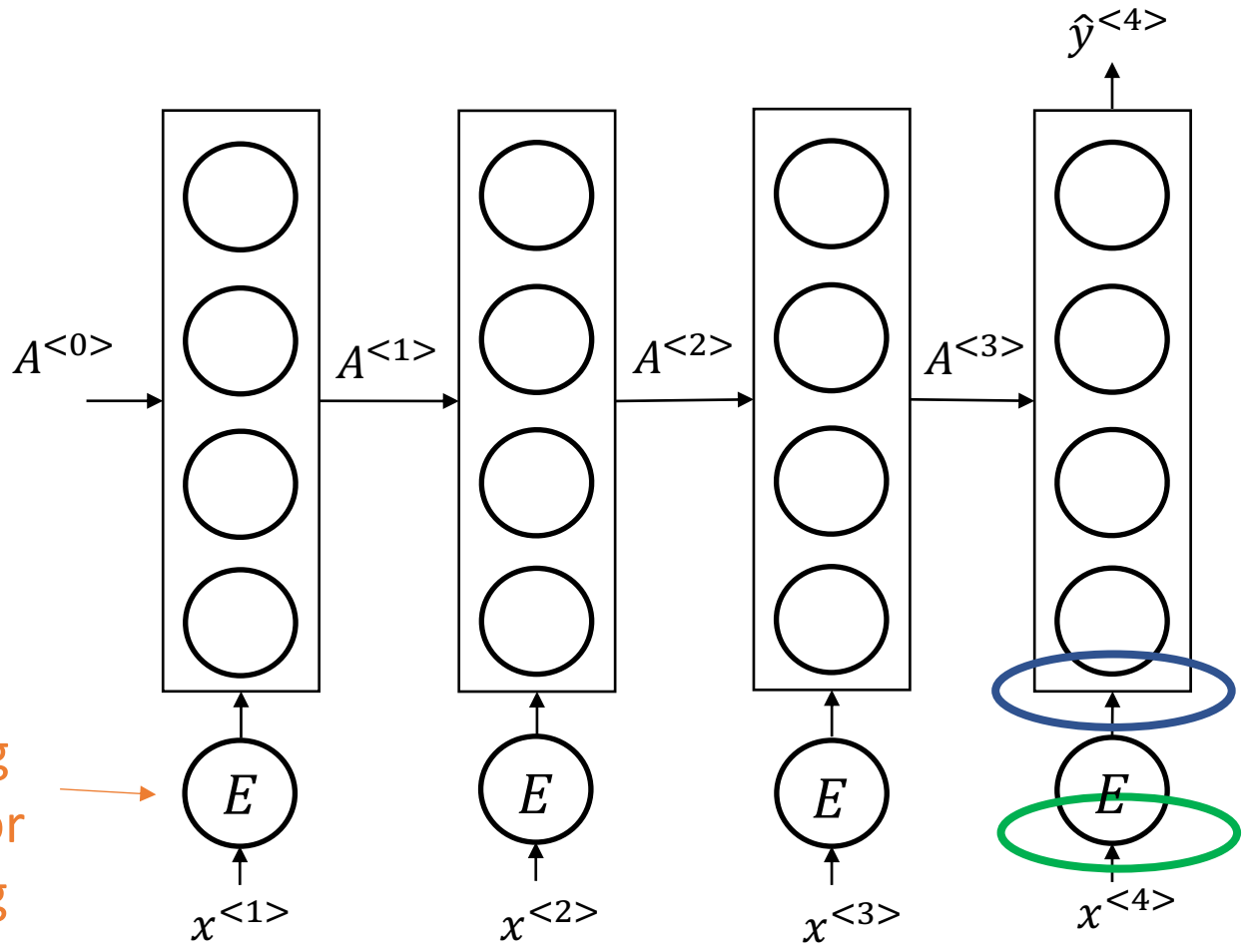
W_{ax} W_{aa} E

We can add an encoding step, by simply multiplying the one-hot-vector by the Embedding Matrix.

Vector size = height of Embedding Matrix (250)

Vector size = length of vocabulary (20,000)

Adding Learnable Embedding Matrix



Learned parameters:

W_{ax} W_{aa} E

(250,20000)

We can add an encoding step, by simply multiplying the one-hot-vector by the Embedding Matrix.

Vector size = height of Embedding Matrix (250)

Vector size = length of vocabulary (20,000)

Embedding Matrix From Language Models

- Now, we can take very large data sets (for instance, all of Wikipedia, or all digital books) and learn an embedding matrix that helps *minimize the cost* of the next word prediction problem

- But, the really exciting thing is that we can remember the values for E and re-use them for other applications

- This is a form of transfer learning. If we create E once on a very large and high-quality data set, we can use it as a starting point for other NLP tasks where we have less example data

Embedding Matrix From Language Models

- The consequences of this is that new applications (like our basketball translation task) do not have to start from scratch!

- For example, the words: *great, excellent*, *exceptional*, will likely all have very similar encodings in the embedding matrix so we are likely to be able to translate all of them successfully even when we do not have training examples for them all in our data set

- Notice that has happened here: *We have learned our implicit human understanding of how words relate to each other from observing their usage…*

Attention Models

- One last topic we will cover in NLP is attention models
- Attention models are quite a new concept in NLP, and are a great example of how quickly Deep Learning changing and progressing as researchers dig deeper into problems
- Let’s take a quick look…

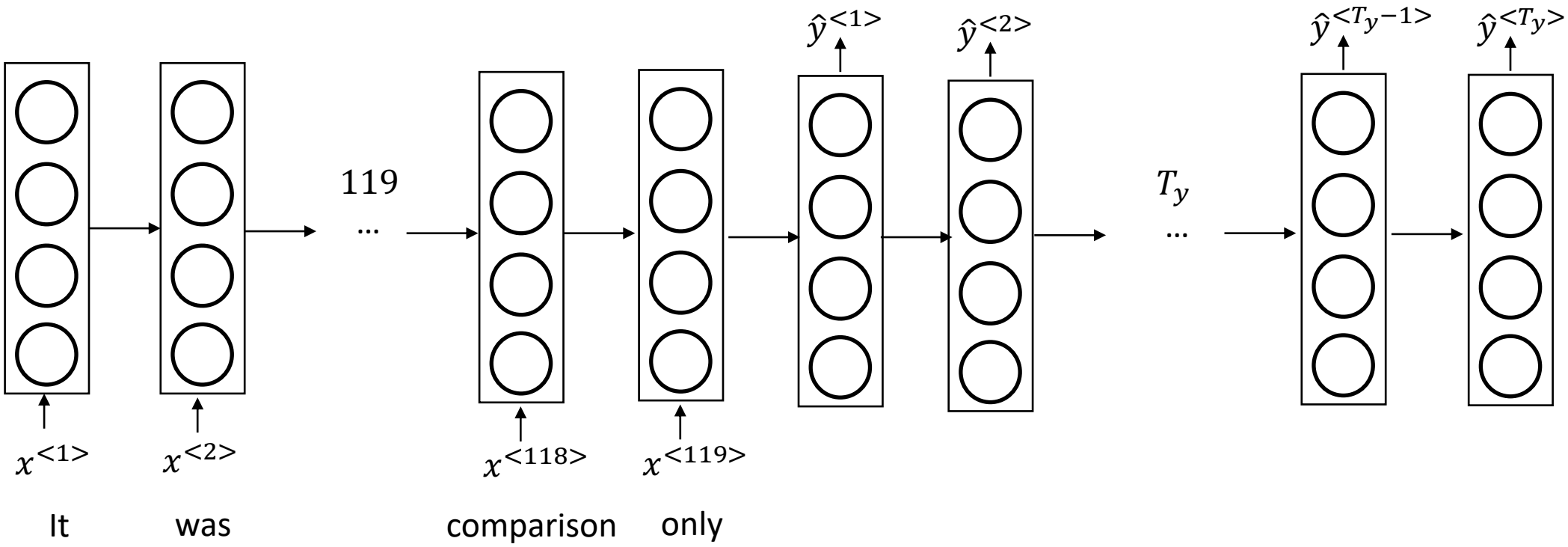
Attention Models

- Consider the problem of translating a very long sentence:

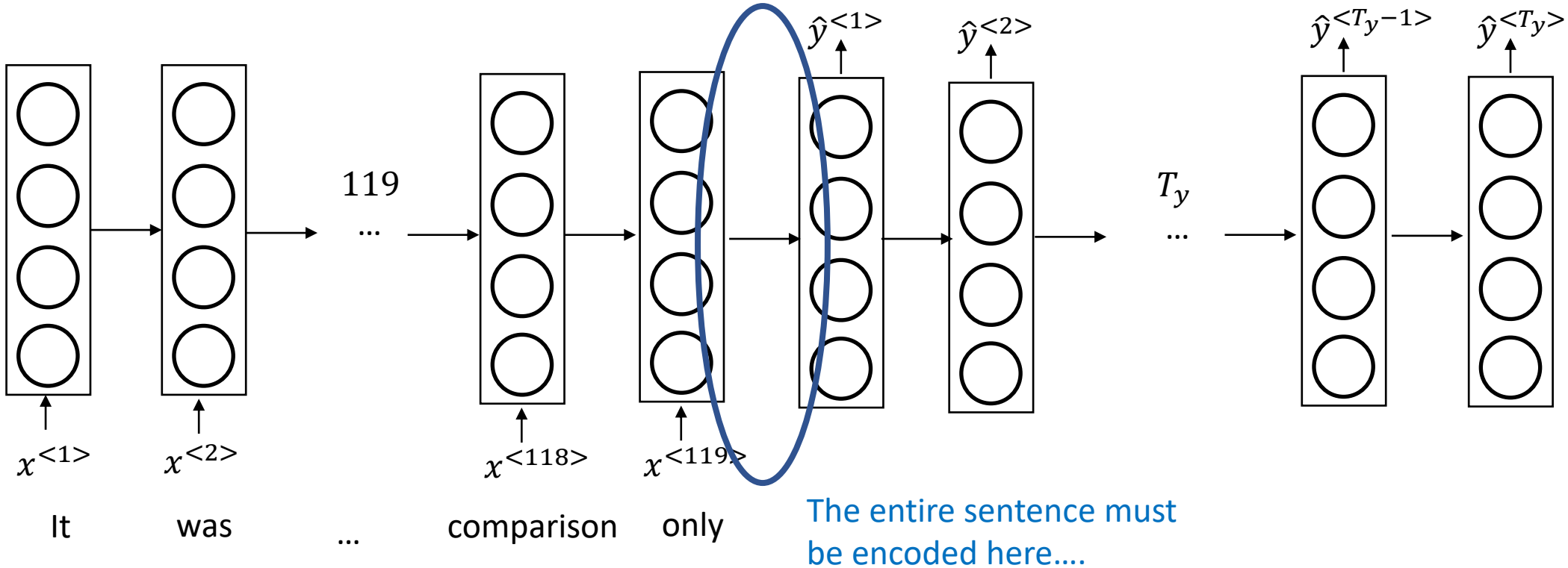
*“It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to Heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.”**

*”A tale of Two Cities” by Charles Dickens.

Machine Translation Example



Machine Translation Example



Attention Models

- What makes intuitive sense is that we would like the output sequence generator to “pay attention” to a selection of the activations of the input words:

- *“It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to Heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.”**

Attention Models

- What makes intuitive sense is that we would like the output sequence generator to “pay attention” to a selection of the activations of the input words:

- “*It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to Heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.”**

Attention Models

- What makes intuitive sense is that we would like the output sequence generator to “pay attention” to a selection of the activations of the input words:

- “*It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to Heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.”**

Attention Models

- What makes intuitive sense is that we would like the output sequence generator to “pay attention” to a selection of the activations of the input words:

- *“It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to Heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.”**

Attention Models

- What makes intuitive sense is that we would like the output sequence generator to “pay attention” to a selection of the activations of the input words:

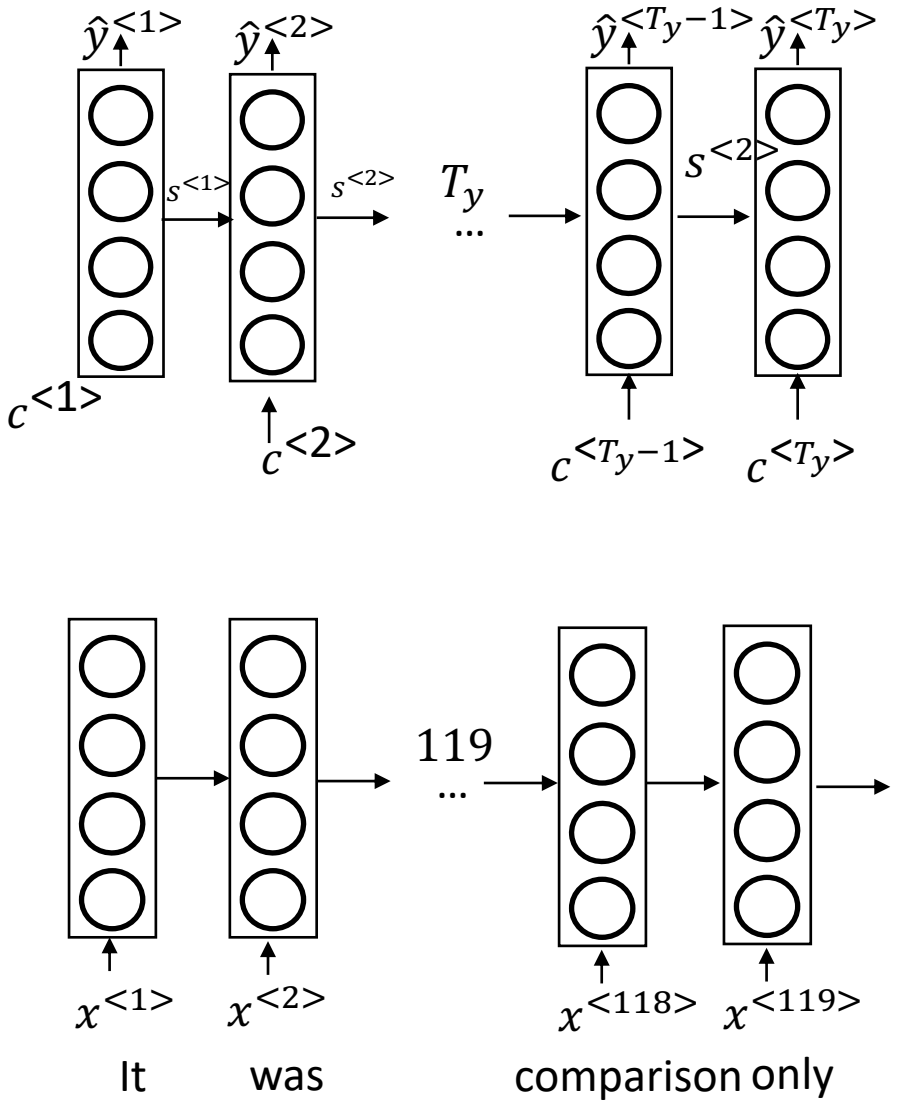
- *“It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to Heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.”**

But, how much should we look at? And what matters most? Maybe we can learn that…

Brad Quinton, Scott Chin

Attention Models

- Let's re-work our model to enable that view

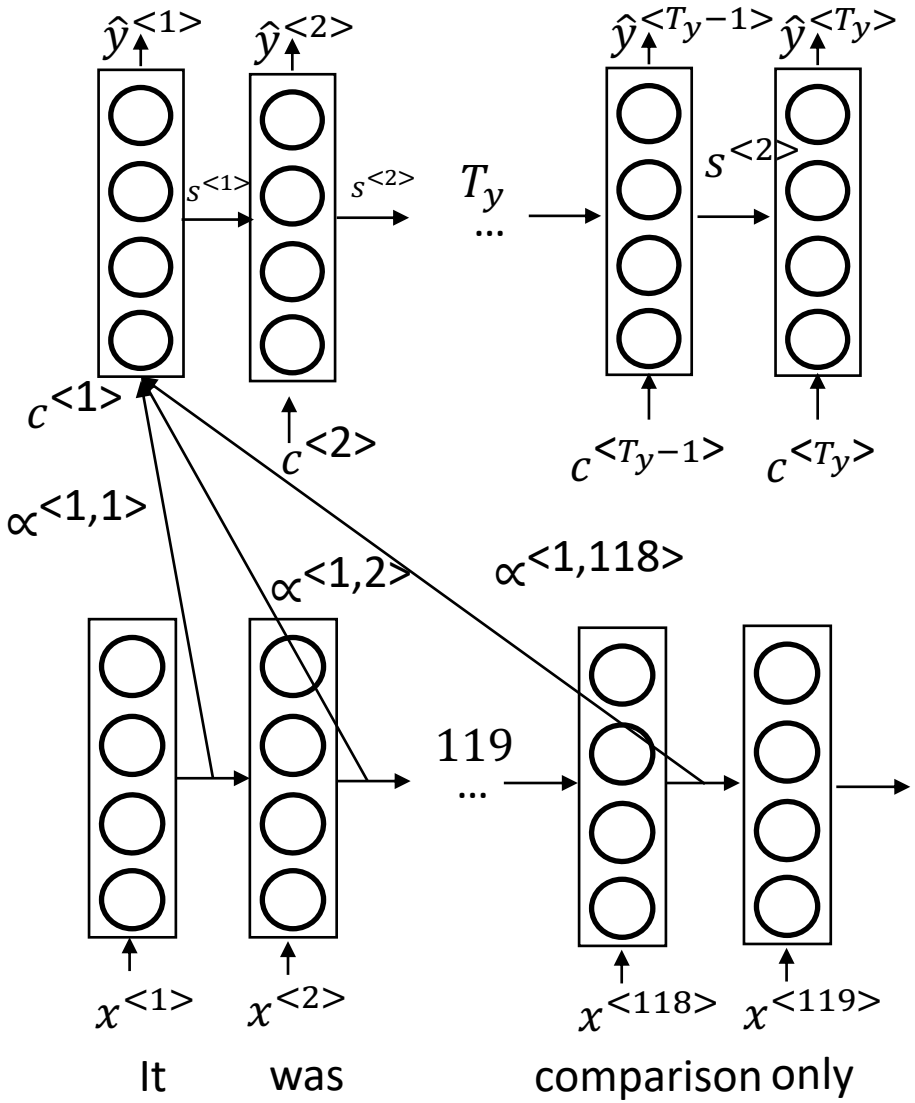


Attention Models

- Let's re-work our model to enable that view

- Lets define $\propto$ as the amount of attention that should be paid to each activation, and define:

$$\sum_1^{t'} \propto^{<1,t'>} = 1$$



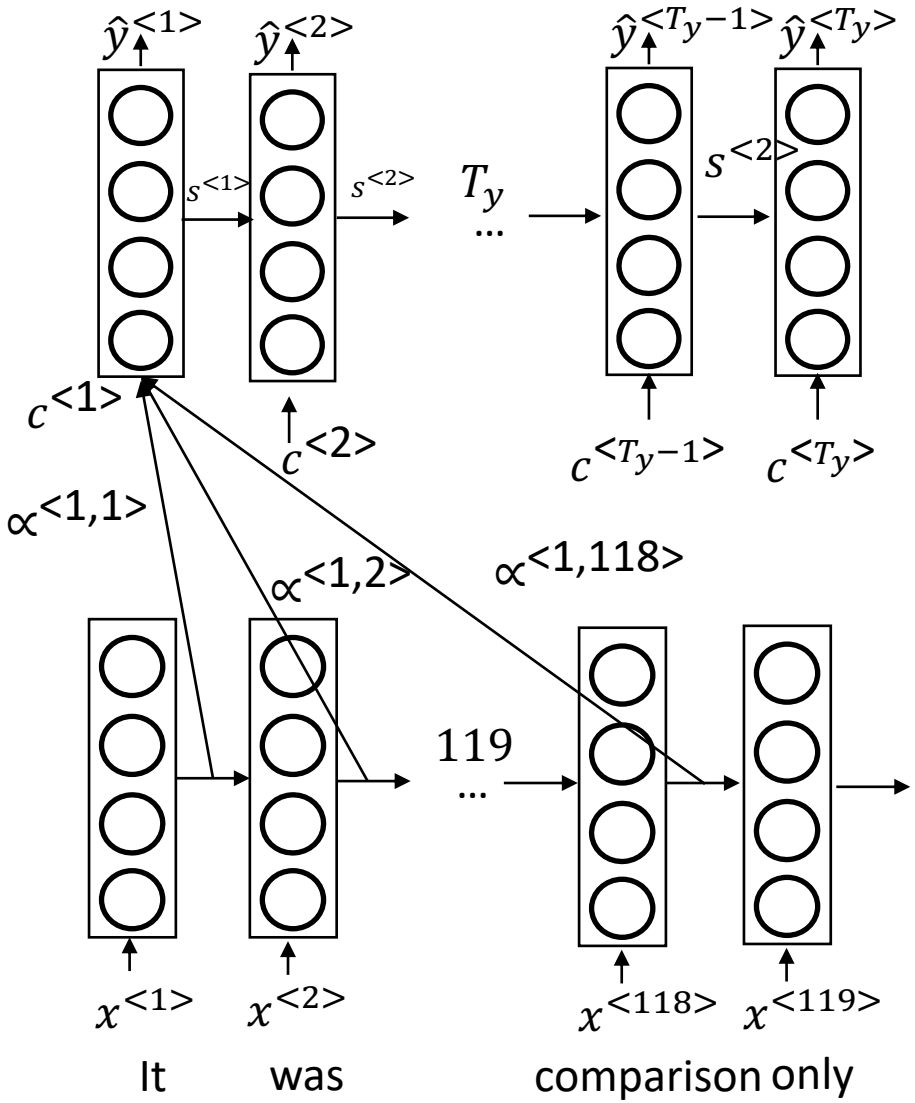
Attention Models

- Let's re-work our model to enable that view
- Lets define $\propto$ as the amount of attention that should be paid to each activation, and define:

$$\sum_1^{t'} \propto^{<1,t'>} = 1$$

- Further, lets define the context for each output sequence to be c:

$$c^{<1>} = \sum_1^{t'} \propto^{<1,t'>} a^{<t'>}$$



Attention Models

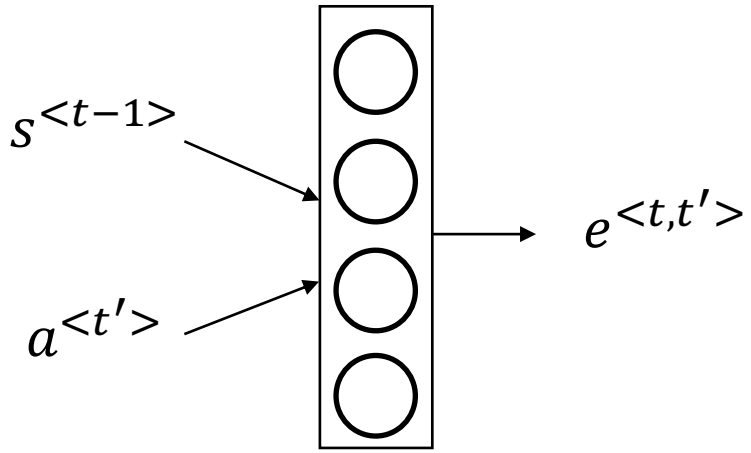
- So now, all we need to do is define a method for computing our attention weights. Remember what we need alphas to sum to 1, so similar to softmax:

$$\propto^{<t,t'>} = \frac{\exp(e^{<t,t'>})}{\sum_1^{T_x} \exp(e^{<t,t'>})}$$

- What about $e^{<t,t'>}$? We can assume that $e^{<t,t'>}$ is a function of the previous activations of the *output* sequence generator and the incoming activation

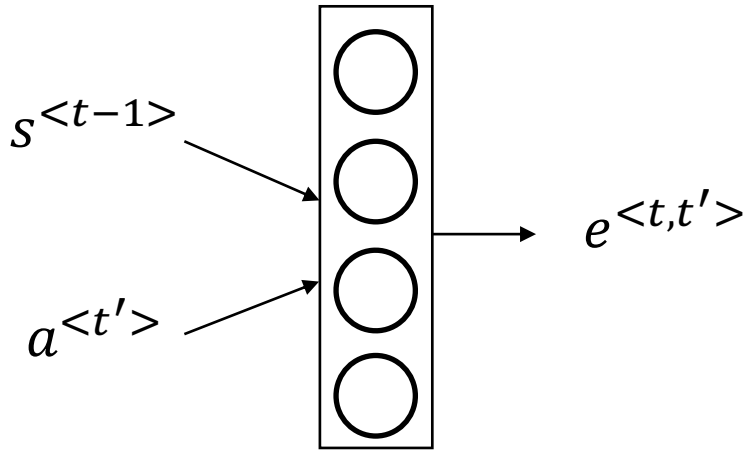
Attention Models

- However, it is not clear exactly what that relationship is, so we can actually use a Neural Network to learn what works best…



Attention Models

- However, it is not clear exactly what that relationship is, so we can actually use a Neural Network to learn what works best…



Now, we can learn each $e^{<t,t'>}$
which will set the value of $\propto^{<t,t'>}$!

Attention Models

- Pulling all that together we can go ahead and setup our learning problem, use gradient descent to drive backprop and, at least for machine translation, get some pretty good results

- ”Under the hood”, what is happening is that by making these changes we are giving the ML system to flexibility to create a function that is well matched to the task at hand

- As we have seen throughout this course, if you can setup the potential to create a function in a way that can leverage backprop to find the parameters, *and* you have good data, you are likely to find a pretty good solution!

Reflecting on What We Have Covered

- Although we have not covered all aspects of NLP, I hope our journey through RNNs, GRUs, LSTM, Word Embedding, and Attention models, has shown you some key things:

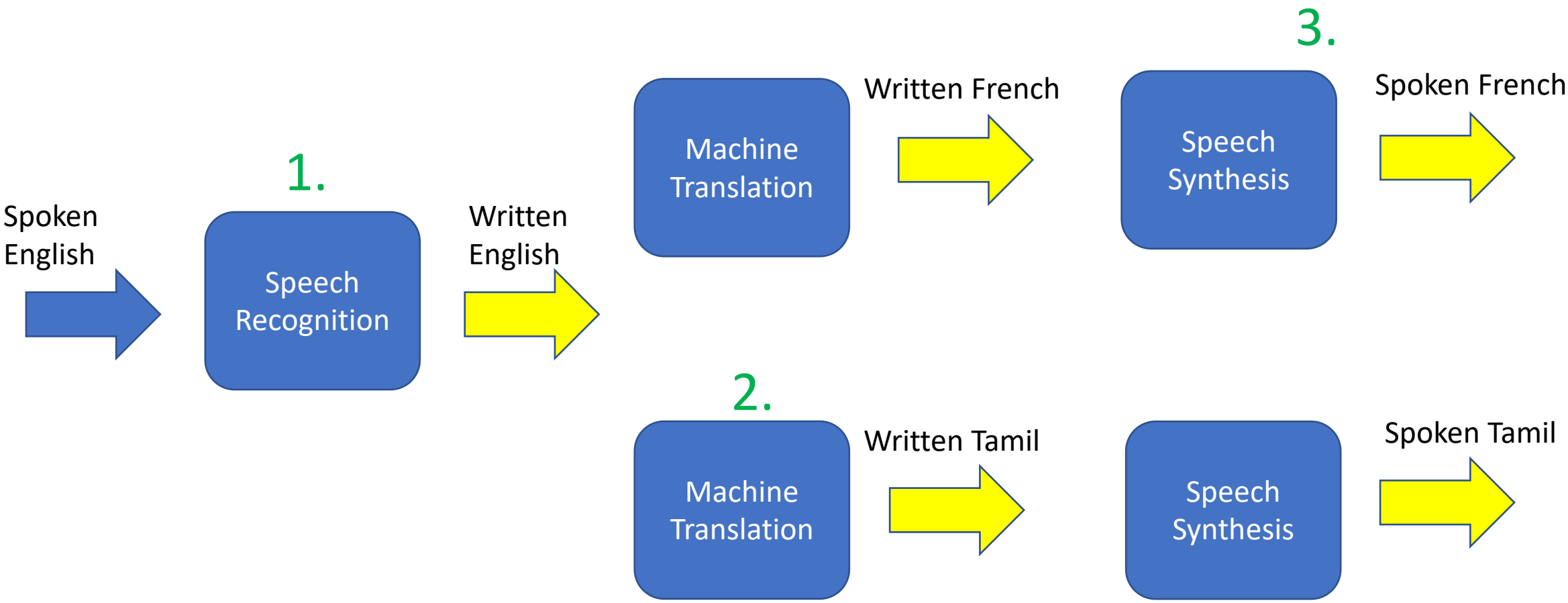
1. Deep Learning and NLP is still in a lot of flux. There are huge opportunities to understand problem spaces and create new ML architectures to attack them. Nothing is really “settled”, yet.

2. By “looking under the hood” in this course and understanding the details of backprop and learning to derive gradient descent on from the basic computation graph, you have the skills to really create these new architectures, as well as understand new architectures that other create.

What About of Case Study?

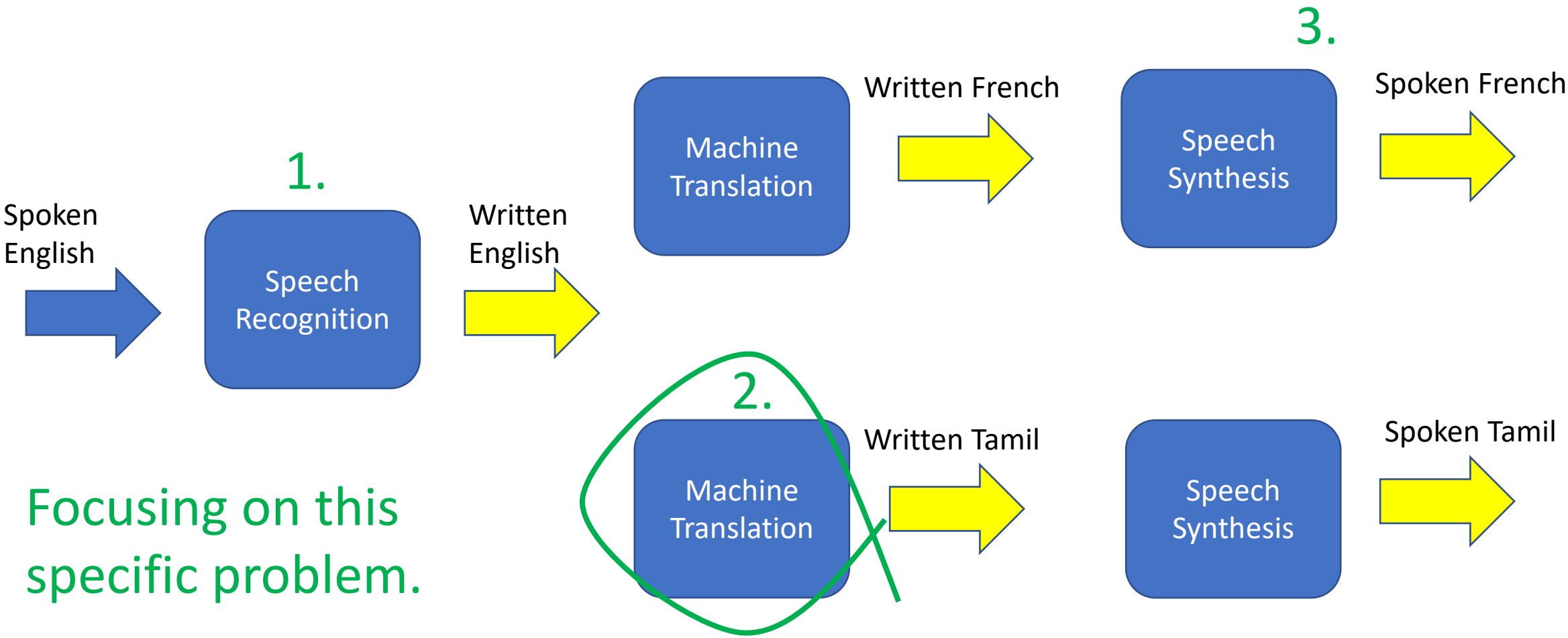
- We shouldn’t forget our original motivation for going down this path.
- What about our simultaneous AI translator for the Toronto Raptors? Are we going to deliver on the CEOs request?
- Let’s sketch out what it would look like to pull it together….

Hypothetical case study….



Brad Quinton, Scott Chin

Hypothetical case study….



Focusing on this specific problem.

Brad Quinton, Scott Chin

Basketball Machine Translator

- **Step 1:** Access high-quality vocabularies/Embedding Matrix for each of the languages we want to address (these are available from Google, or OpenAI, etc.)

- **Step 2:** Find as many translated basketball game transcripts as possible. (In our case we would likely have to hire some human translators.)

- **Step 3:** Split the data we do collect into training, validation and test data sets.

- **Step 4:** Use a framework like TensorFlow, Keras or PyTorch to create a Bi-direction LSTM network with Attention.

Brad Quinton, Scott Chin

Basketball Machine Translator

- **Step 5:** Train our network to do sentence by sentence translation with the words acting as the sequential elements in our data.
- **Step 6:** Tune the hyperparameters using the validation data set.
- **Step 7:** Test the tuned network on the test data set to evaluate its effectiveness.
- **Step 8**: If we are satisfied with the test accuracy, **build a deployment**!

Course Evaluations

- Course evaluations are now open!
- Scott and I would really appreciate if you could spend the time to do these evaluations
- As you know, this is a newly developed course. Your feedback is important to us, if you have places we can improve, please be detailed and specific!