Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection

What is Paraphrase Detection?

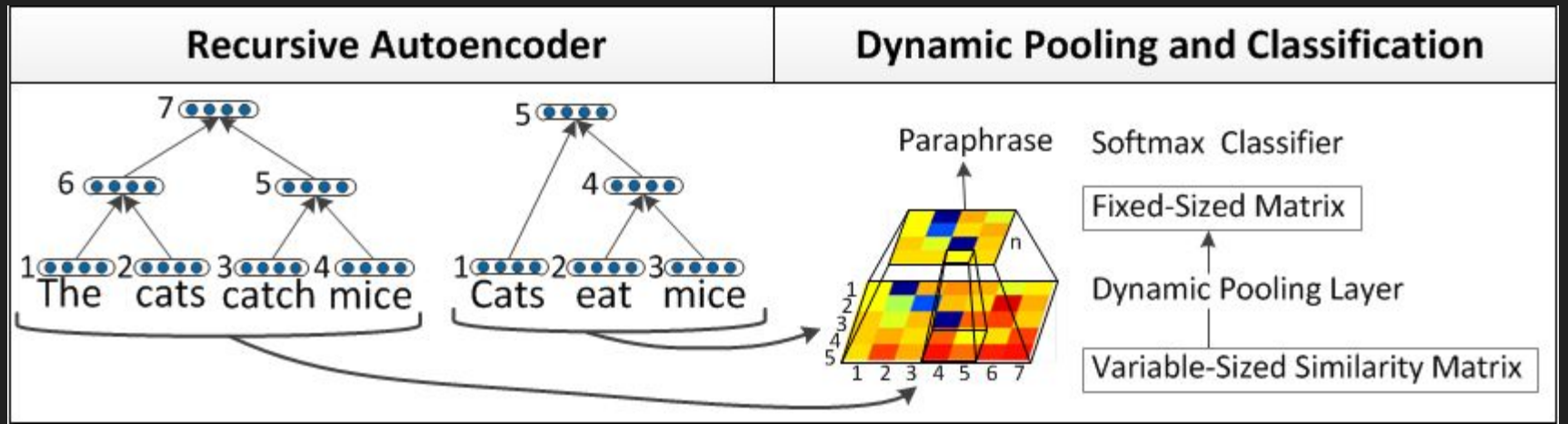
- “Paraphrase detection determines whether two phrases of arbitrary length and form capture the same meaning”
  - Basically, we want to compute the similarity between two sentences in meaning
- Why are we trying to teach computers how to do this?

What is Paraphrase Detection?

- “Paraphrase detection determines whether two phrases of arbitrary length and form capture the same meaning”
  - Basically, we want to compute the similarity between two sentences in meaning
- Why are we trying to teach computers how to do this?
  - Information retrieval/Question answering
  - Text summarization/Plagiarism detection
  - Evaluation of machine translation

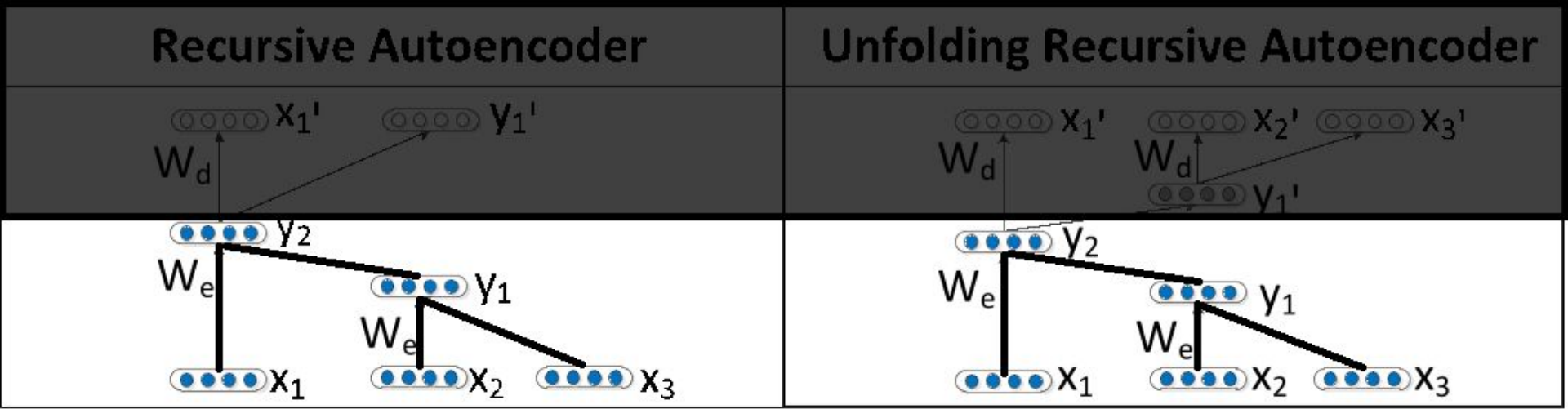
The General Idea

- Create a feature vector for both sentences 1 and 2
  - These vectors store *relationships* between words in the sentence
- Put these two vectors together into a matrix
  - This matrix now has the word relationships stored in a 2D format



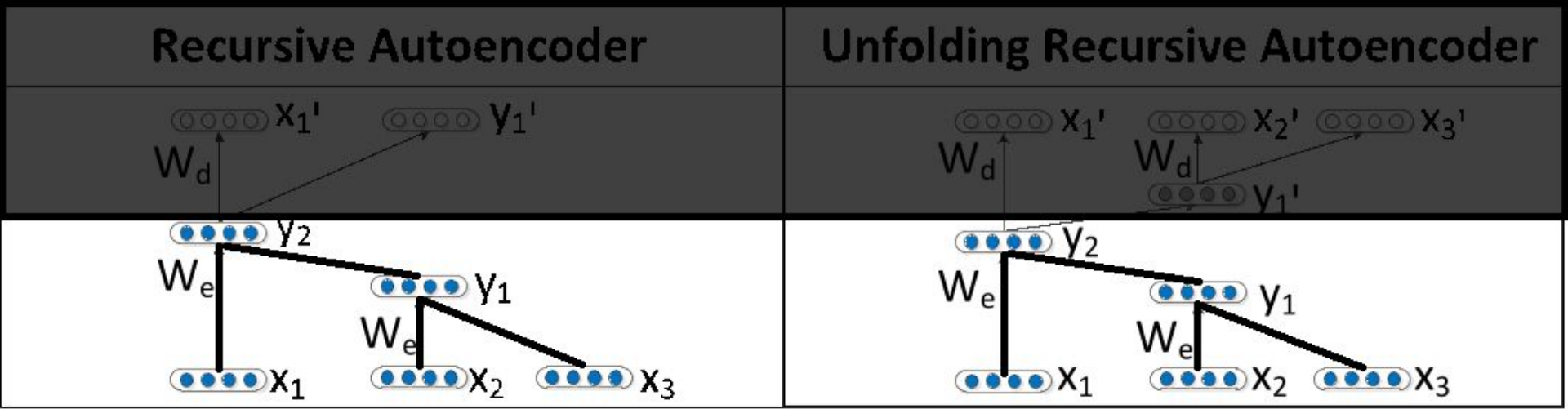
Recursive Auto-Encoder General Idea

- This paper is neat because it does not use LSTM's to do NLP
- Auto encoders to encode sentence meaning
  - Will spend most of our time on this
- Use pre-existing *Syntactic Parser* to create a sentence tree



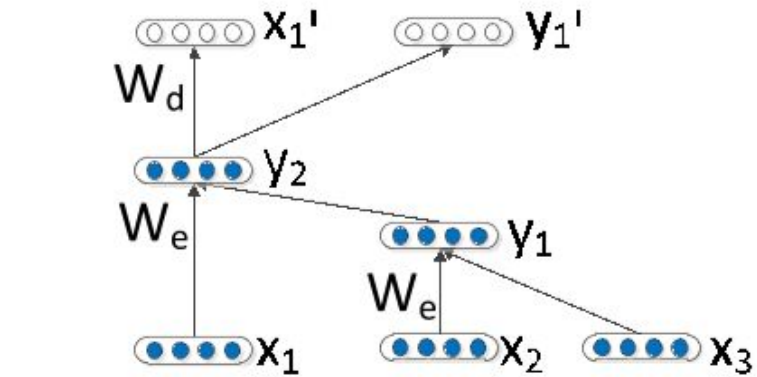
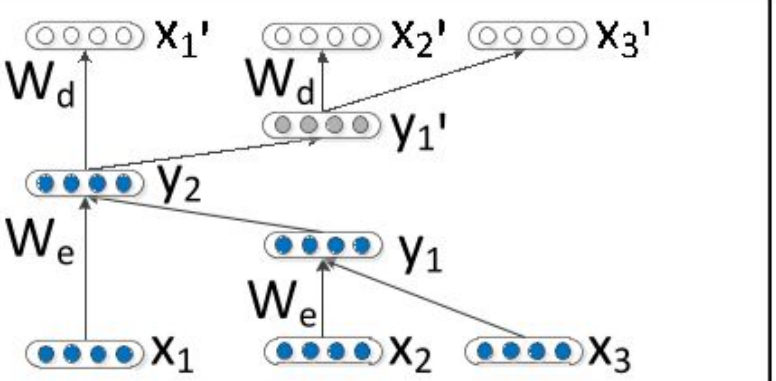
Sentence Trees

- Syntactic parser finds parent-child relationships between words
- y_1 is parent of x_2, x_3; y_2 is parent of x_1, y_1
  - All of these variables are vectors! (of length 200)
- Our auto-encoder uses the structure of the syntactic tree to recursively compress the vectors



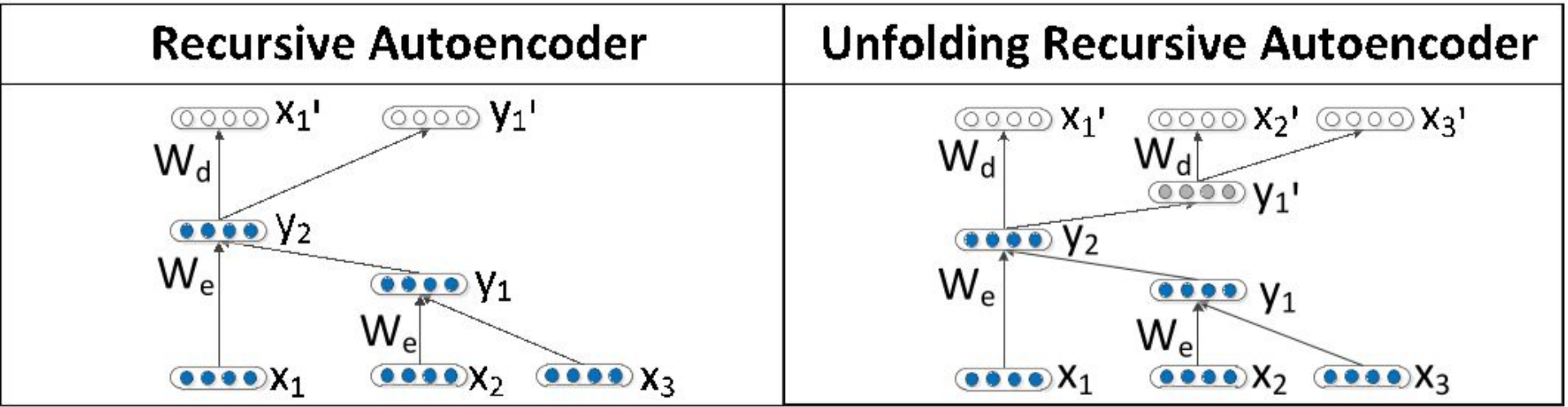
Encode, but how good is the encoding?

- y_2 contains an encoded version of the sentence
- How can we tell how good our encoded version is?
  - Euclidean distance for error = $$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1-p_1)^2+(q_2-p_2)^2+\cdots+(q_n-p_n)^2} = \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}.$$

Recursive Autoencoder	Unfolding Recursive Autoencoder
	

Some encoding issues to consider

- What if y_n is the encoding of 10 words and is paired with x_{11}?
  - How do we make sure the error is spread out accordingly?

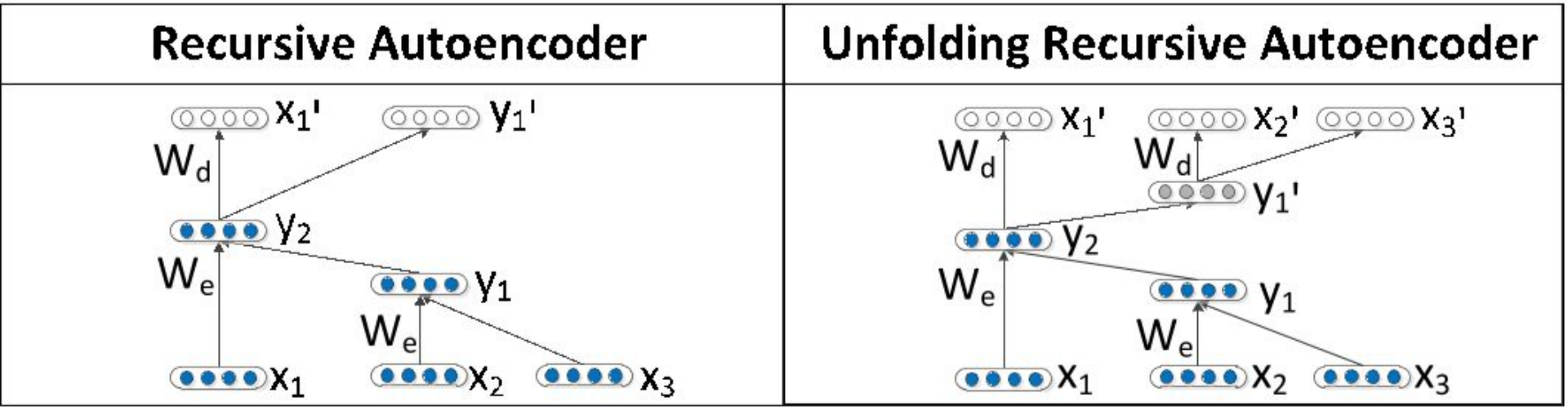


Some encoding issues to consider

- What if y_n is the encoding of 10 words and is paired with x_{11}?
  - How do we make sure the error is spread out accordingly?

For $y_{(i, j)}$ consisting of words x_i through x_j,

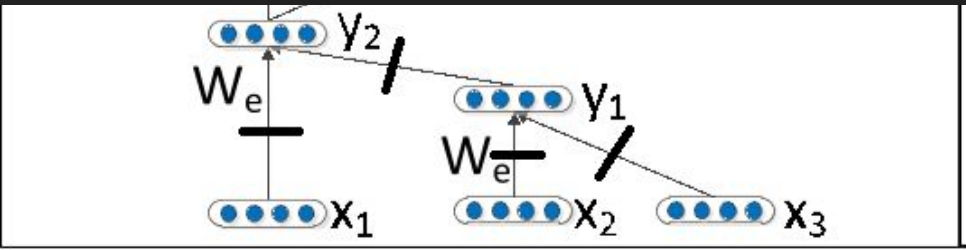
$$E_{rec}(y_{(i,j)}) = \left|\left|[x_i; \ldots; x_j] - [x'_i; \ldots; x'_j]\right|\right|^2$$

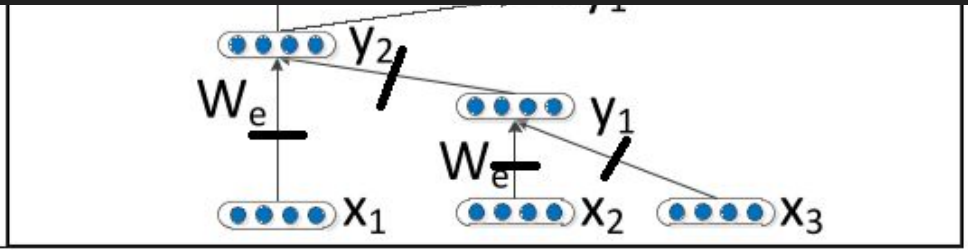


Gotta go deeper

- Is one layer of encoding enough?
- Will our dimensions be preserved?
  - Will x's and y's have same dimensions?

$$
\begin{aligned}
h &= f(W_e^{(1)}[c_1; c_2] + b_e^{(1)}) \\
p &= f(W_e^{(2)}h + b_e^{(2)}).
\end{aligned}
$$





Results of encoding and decoding

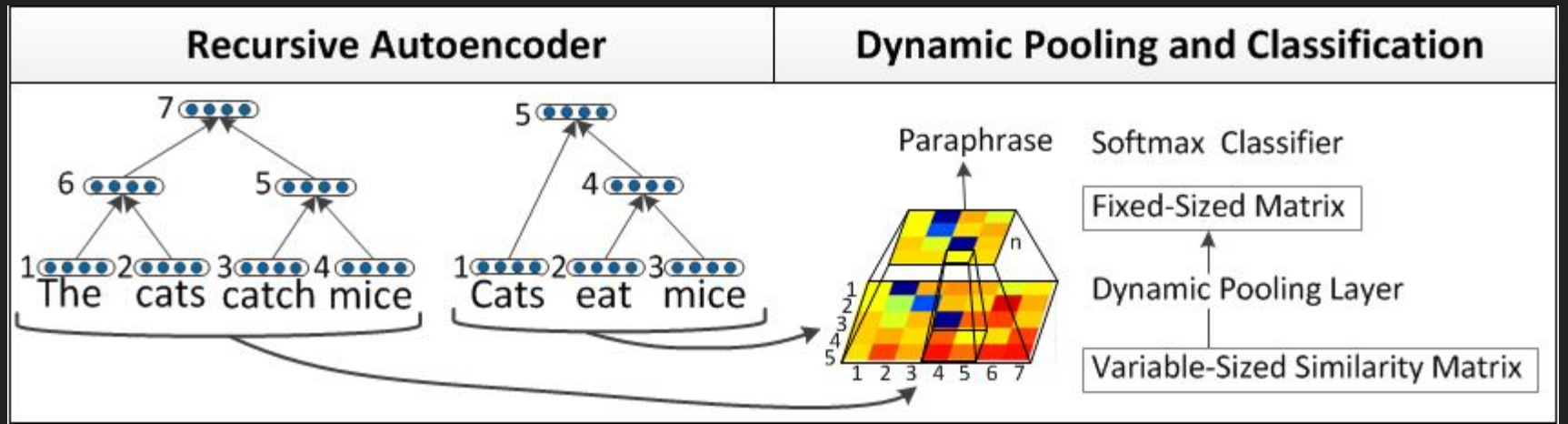
Nearest neighbor means that they took the word that is closest by Euclidean distance to the vector after reconstructing

Encoding Input	Generated Text from Unfolded Reconstruction
a December summit	a December summit
the first qualifying session	the first qualifying session
English premier division club	Irish presidency division club
the safety of a flight	the safety of a flight
the signing of the accord	the signing of the accord
the U.S. House of Representatives	the U.S. House of Representatives
enforcement of the economic embargo	enforcement of the national embargo
visit and discuss investment possibilities	visit and postpone financial possibilities
the agreement it made with Malaysia	the agreement it made with Malaysia
the full bloom of their young lives	the lower bloom of their democratic lives
the organization for which the men work	the organization for Romania the reform work
a pocket knife was found in his suitcase in the plane's cargo hold	a bomb corpse was found in the mission in the Irish car language case

Table 2: Original inputs and generated output from unfolding and reconstruction. Words are the nearest neighbors to the reconstructed leaf node vectors. The unfolding RAE can reconstruct perfectly almost all phrases of 2 and 3 words and many with up to 5 words. Longer phrases start to get incorrect nearest neighbor words. For the standard RAE good reconstructions are only possible for two words. Recursive averaging cannot recover any words.

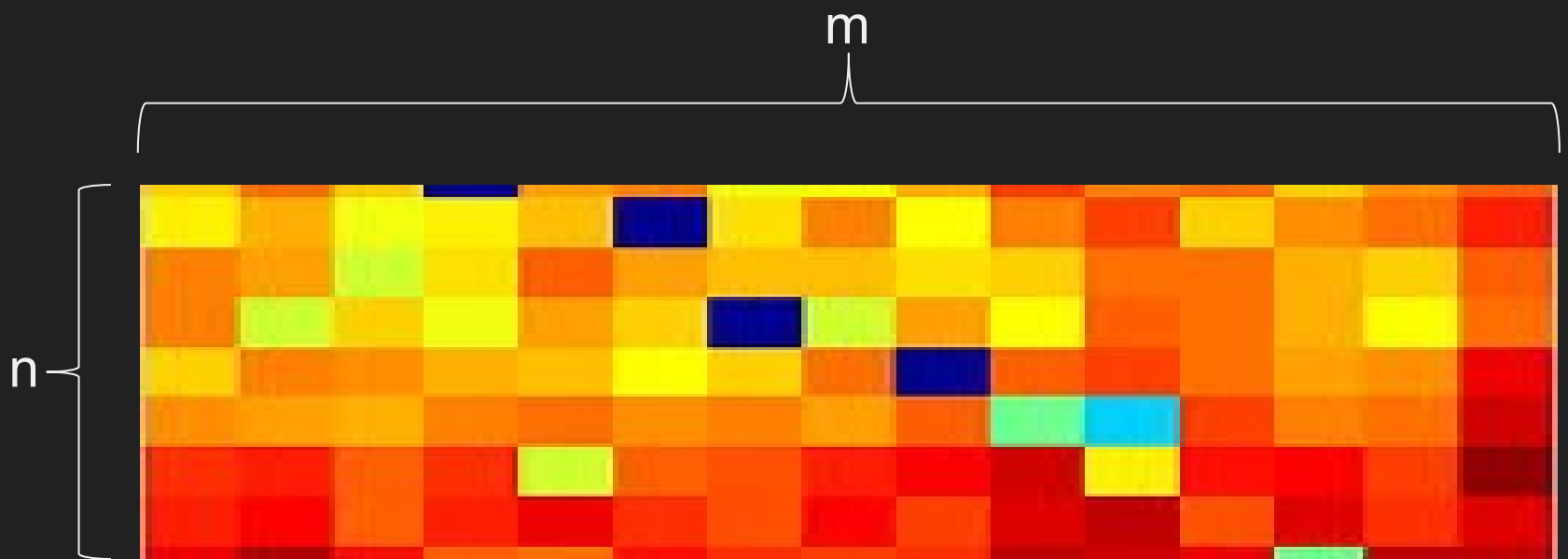
We've encoded the sentences, now what?

- Put them into a matrix of size s_n x s_m
- Why do we put all the vectors into the matrix rather than just the encoding?
  - Including the original $x_{1\text{-}k}$ word vectors necessary?



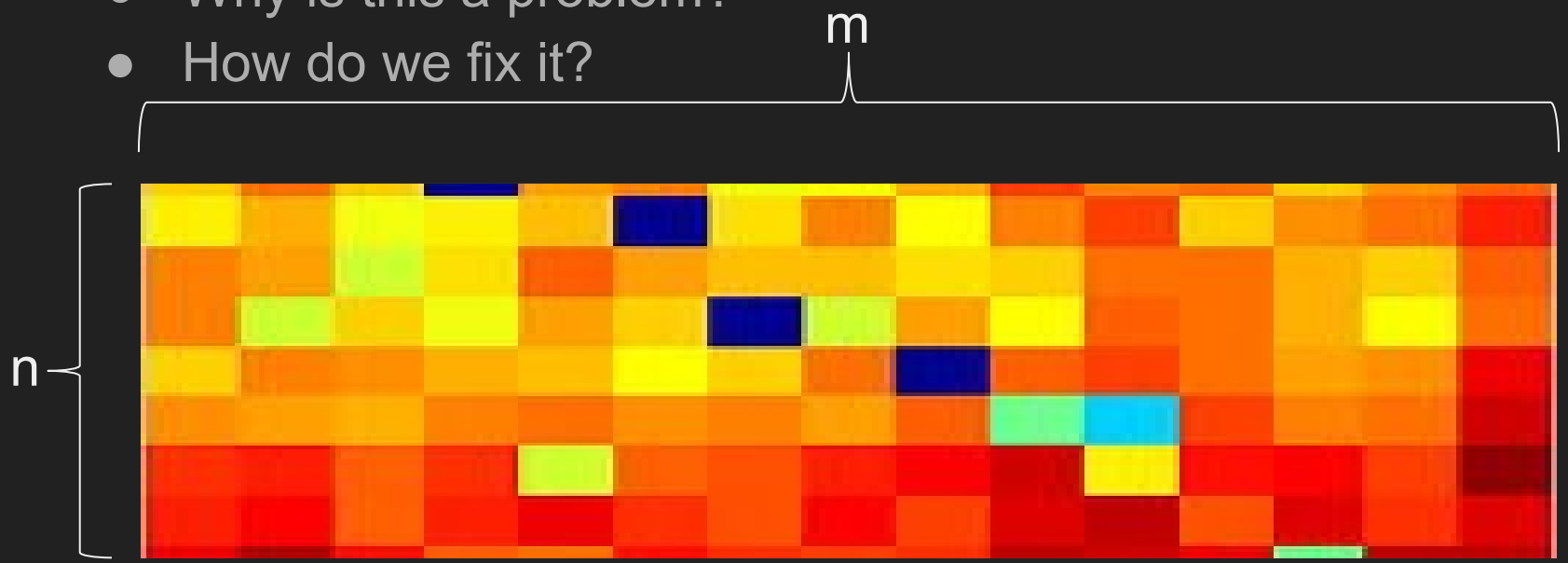
What do we fill the matrix with?

- n and m are the numbers of feature vectors from sentences 1 and 2
- Does it matter what order the x's and y's have in the matrix?
  - x and y being the word/encoding vectors from 2 slides ago



Our matrix dimensions may be different

- (We've filled the matrix with Euclidean distances)
- Why is this a problem?
- How do we fix it?



Min pooling

- Split the matrix into p x p sections
- Take the minimum value from each section
  - But max-pooling in convolutional networks does max value, why min value here?
  - Do we lose information?

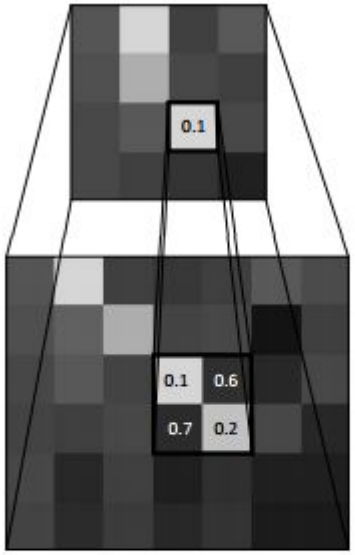


Figure 3: Example of the dynamic min-pooling layer finding the smallest number in a pooling window region of the original similarity matrix $\mathcal{S}$.

We have the matrix, now what do we do?

- Just one layer of a neural network
  - What is the input to this neural network?
  - What is the output?
- Are we done?

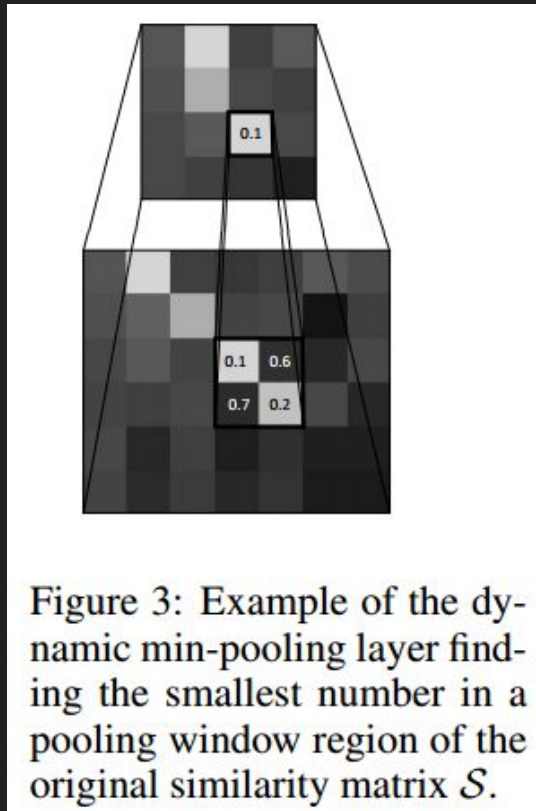


Figure 3: Example of the dynamic min-pooling layer finding the smallest number in a pooling window region of the original similarity matrix $\mathcal{S}$.

A few small additions

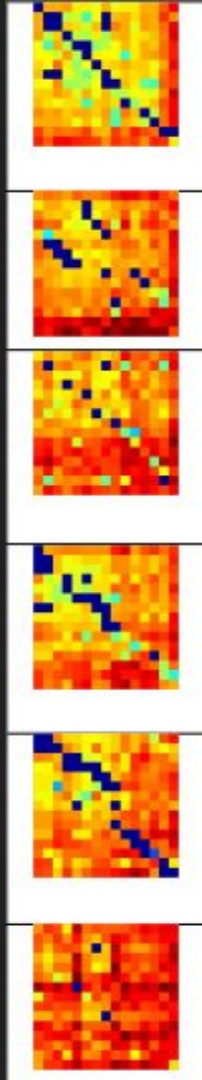
- The researchers noticed that different numbers in the sentences would be encoded similarly, but could have a big impact on a lack of paraphrasing
- To account for this, they added a few variables
  - V_1 = 1 if both sentences have exactly the same numbers, 0 otherwise
  - V_2 = 1 if two sentences contain same numbers
  - V_3 = 1 if one sentence’s numbers are subset of other sentence’s numbers
- Where do we add these new variables to our system?

A few small additions

- The researchers noticed that different numbers in the sentences would be encoded similarly, but could have a big impact on a lack of paraphrasing
- To account for this, they added a few variables
  - (Not specified in the paper, but this is my best guess)

The resulting pooled matrices

- Blue = small euclidean distance, Red = large euclidean distance
- Why is the diagonal more blue?
  - Why more red when farther from diagonal?



Let’s go over what we did

Wanted to detect probability of sentence similarity

1. We encode the two sentences down to a single vector, and get a few other vectors along the way
2. Put all of these encodings and original word vectors into a matrix which now contains pairwise relationships between sentence features
3. Standardize the size of the matrix
4. Compute sentence similarity based on matrix values

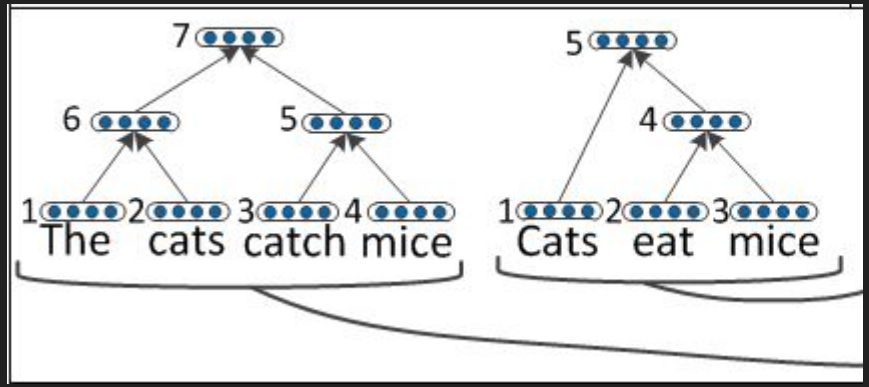
Steps 1 and 4 contain the actual neural network aspects

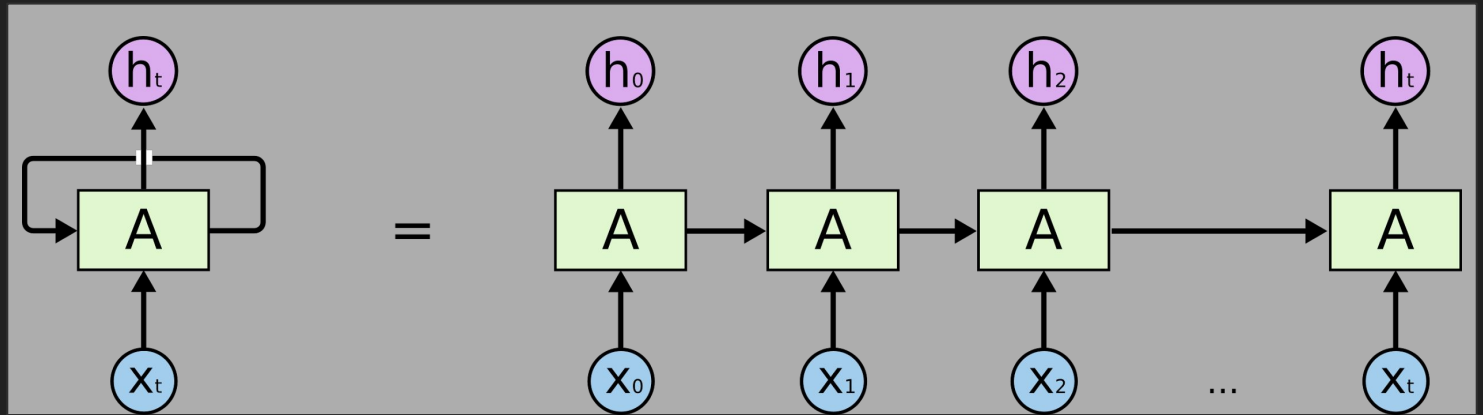
Are the neural networks here complicated?

Are the neural networks here complicated?

- Not particularly
- But they work in tandem with the other algorithms to provide a reliable, state-of-the-art answer
  - Don’t always need ultra-hard deep learning models to achieve good results

Are the encoders/decoders recurrent neural networks?





Some of the sub-algorithms

- Tracking tree recursion while both encoding and decoding
- Splitting matrix into p x p segments
  - What if it doesn’t evenly split?
  - They used p = 15