W2-L1 Word-embeddings

PAPER
Van der Maaten & Hinton 2008. Visualizing data using t-SNE

Wordembeddings can learn feature vector for words. So instead of using one-hot feature vector for words, we ~~would~~ can use feature vectors for words from word-embedding algorithms.

W2-L2

- FACE ENCODING (from CNN) is similar to word embeddings

W2-L3

PAPER: Mikolov et al 2013, Linguistic regularities in continuous space word representation

↑ surprising results for wordembeddings

W2-L4

- When you learn word embedding, you end-up learning embedding ~~vector~~ matrix.

Embedding Matrix $\in R^{300\times|V|}$: 300D vector for word
$|V|$ is total number of words in dictionary

W2-L5 Learning word embeddings

Paper Bengio et al. 2003, A neural probabilistic language model

→ Neural language Model

For language model: You can take context (last 4-words) & build language model.

For word embedding: You may want to pick last & future words to learn meaningful word embedding.

word2 vec

Paper: Mikolov et al. 2013: Efficient estimation of word representation in vector space

Word2VEC ~~SkipGram Model~~:

- → SkipGram Model (problem is denominator)
- → CBOW (picks surrounding words for learning target word)

(write notes from 06:40)

GOAL How to learn word embeddings. lets take a look at SkipGram model.

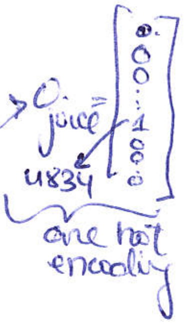
Sentence: "I want a glass of orange juice to go along with cereal"

lets pick ~~target~~ contex word ~~[illegible]~~ & then its target from window of length "t" words

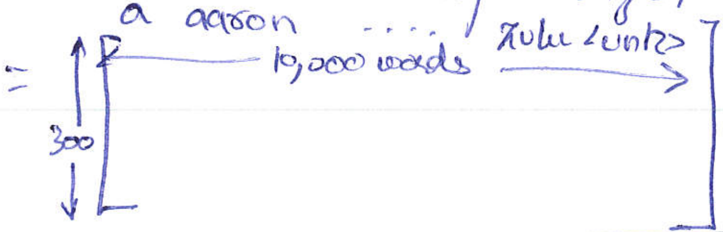
Context	target
orange	juice

Let say Vocab size = 10,000

TASK Context "c" ("orange") 6257 ⟶ Target "t" ("juice") 4834 → O_{juice} =



Here E = embedding Matrix
= 300 x 10,000 (Dim of word embedding X Vocab size)
=



Model:

$$\underbrace{O_c}_{\text{one-hot for context}} \longrightarrow E \longrightarrow \underbrace{e_c}_{\substack{e_c = EO_c \\ \text{embedding vectors}}} \longrightarrow \underset{\text{softmax}}{\bigcirc} \longrightarrow \hat{y}$$

Softmax:

$$P(\text{target}|\text{context}) = \frac{e^{\theta_t^T e_c}}{\sum_{j=1}^{10,000} e^{\theta_j^T e_c}}$$

θ_t = parameters associated with output "t"

$$\mathcal{L}(\hat{y}, y) = -\sum_{i=1}^{10,000} y_i \log \hat{y}_i$$

$$y = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \rightarrow 4834 \text{ (target) = "juice"}$$

Problem

Denominator of softmax $= \sum_{j=1}^{10,000} e^{\theta_j^T e_c}$

It is solved by hierarchical softmax. Have a look for more info. This is one idea to speed up softmax.

W2-L7 Negative Sampling

Papes: Mikolov et.al 2013. Distributed representations of words & phrases and their compositionality

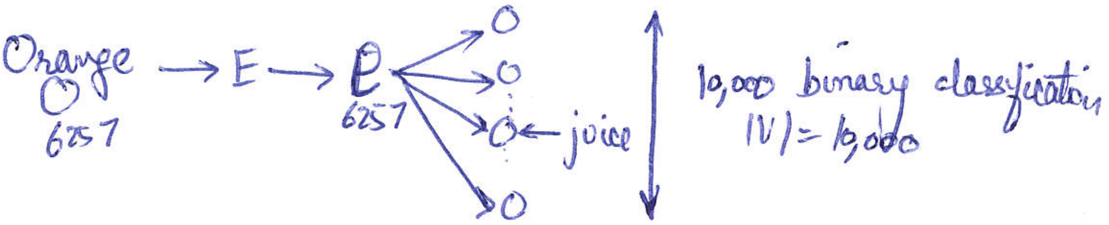
In this lecture:

- Context/target words are training input and $y=1$ if context/target are true pairs

Context	target	y
orange	juice	1
orange	King	0

- Negative samples are sampled from corpus
- logistic regression is applied for ~~ea~~ word pairs

$$P(y=1 \mid c=\text{Orange}, t=\text{juice}) = \sigma(\theta_t^T e_c)$$



W2-L8 Glove Word vectors

↳ Another wordembedding algorithm

Pennington et al 2014 Glove: Global vectors for word representation

W2-L9 Sentiment Classification

- Simple NN
- RNN

} for sentiment classification

W2-L10

Papes: Bolukbasi et al 2016: Man is to computer programmer as woman is to homemaker? Debiasing word embeddings

SEQUENCE to sequence model

PAPER 1) Sutskever et al. 2014. Sequence to sequence learning with neural network

2) Cho et al. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation

$x^{<1>}$ Jane $x^{<2>}$ visite $x^{<3>}$ l'Afrique $x^{<4>}$ en $x^{<5>}$ septembre

→ Jane is visiting Africa in September

$y^{<1>}$ . . . $y^{<6>}$

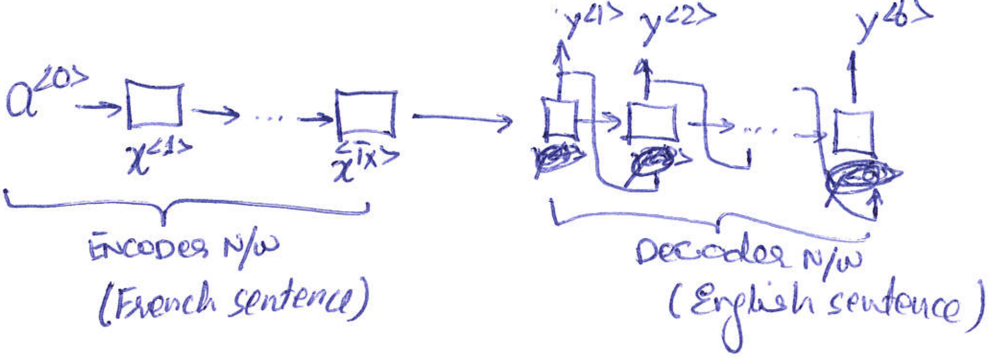
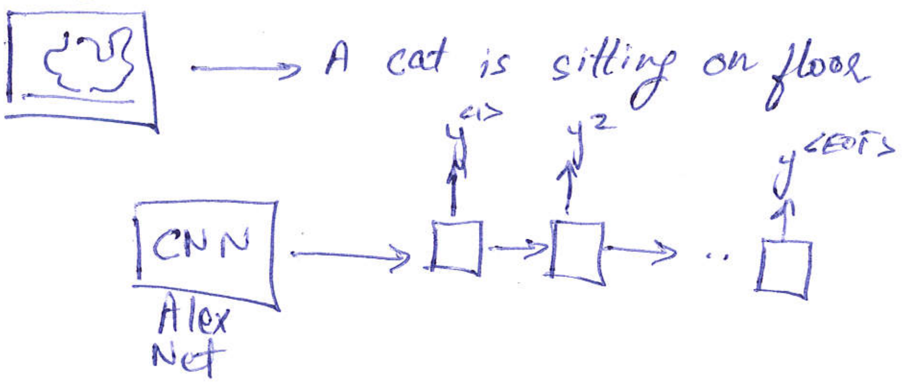


Image captioning



PAPERS 1) Mao et. al 2014. Deep captioning with multimodal RNN

2) Vinyals et. al 2014. Show & tell: Neural image caption generator

3) Karpathy & Fei Fei 2015: Deep visual-semantic alignments for generating image descriptions

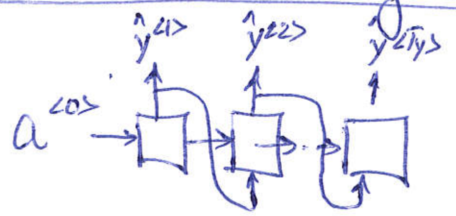
W3-L2

Picking the most likely sequence

In machine translation, we want to pick the most likely sequence. In language model, we sampled words randomly.

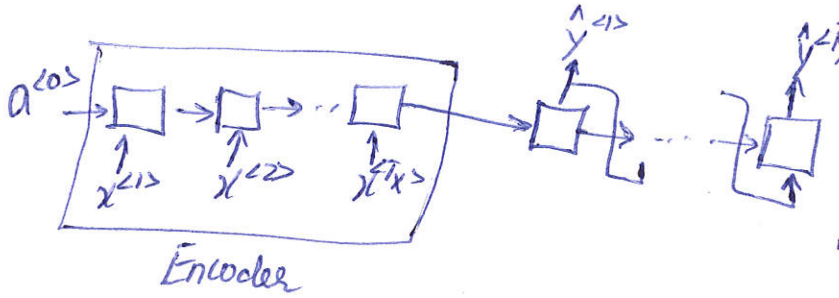
Machine translation as building a conditional language Model

Language model:



Here we were calculating $P(y^{<1>}, \dots y^{<T_y>})$

Machine translation:



Here we want $P(y^{<1>} \dots y^{<T_y>} / x^{<1>} \dots x^{<T_x>})$

- To find the best sequence $P(y^{<1>}, \dots, y^{<T_y>} / x)$, we will use beam search algorithm.

W3-L3 **Beam Search** (see the video)

W3-L4 **Refinements to Beam search**

In beam search, we would like to get the most likely sequence (y) such that

$$P(y^{<1>}, y^{<2>}, \dots y^{<T_y>} | x) = \underset{y}{\arg\max} \Big[P(y^{<1>}|x) \times P(y^{<2>}|x, y^{<1>}) \times P(y^{<3>}|x, y^{<1>}, y^{<2>}) \dots \times P(y^{<T_y>}|x, y^{<1>}, \dots y^{<T_y-1>}) \Big]$$

$$= \underset{y}{\arg\max} \prod_{t=1}^{T_y} P(y^{<t>} | x, y^{<1>} \dots y^{<t-1>})$$

Refinments to beam Search

1. Length normalization

maximize log probabilities to avoid underflow

2.) $\underbrace{\frac{1}{T_y^{\alpha}} \sum_{t=1}^{T_y} \log P(y^{<t>} | x, y^{<1>}, \dots, y^{<t-1>})}_{\text{Normalize log-likelihood objective}}$ $\alpha = [0, 1]$

W3-25 Error-Analysis on beam Search

↳ who is at fault

1) Beam Search or 2) RNN ?

W3-26 BLEU score (Bilingual evaluation understudy)

Paper: Papineni et al. 2002. A method for automatic evaluation of machine translation $\boxed{\text{(Very influential)}}$

BLEU Score → for machine translation
→ for captioning system
↳ can be used when text is generated and multiple good text is available as ~~given~~ good text prediction

W3-27 Attention Model Intuition

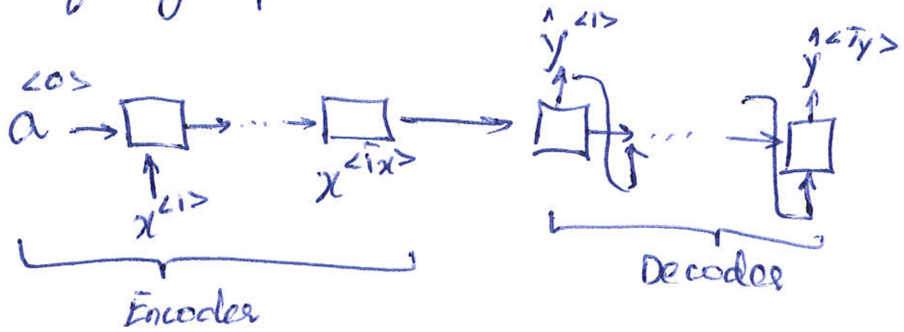
Paper: Bahdanau et al 2014. Neural machine translation by jointly learning to align & translate.

W3-28 Attention_Model

Paper: 1) Bahdanau et.al 2014. Neural machine translation ...
2) Xu et.al. 2015. Show attention and tell: neural image caption generation with visual attention

45-a) Attention Model Intuition

The Problem of long Sequences



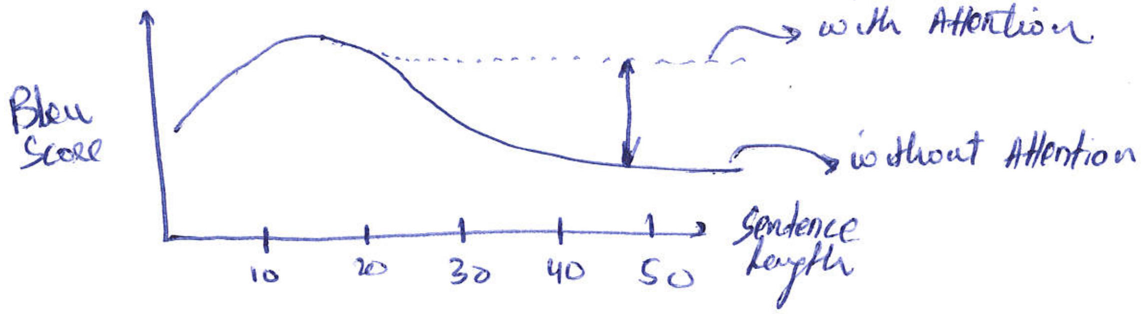
French Jane s'est rendue en Afrique en .... aussi.

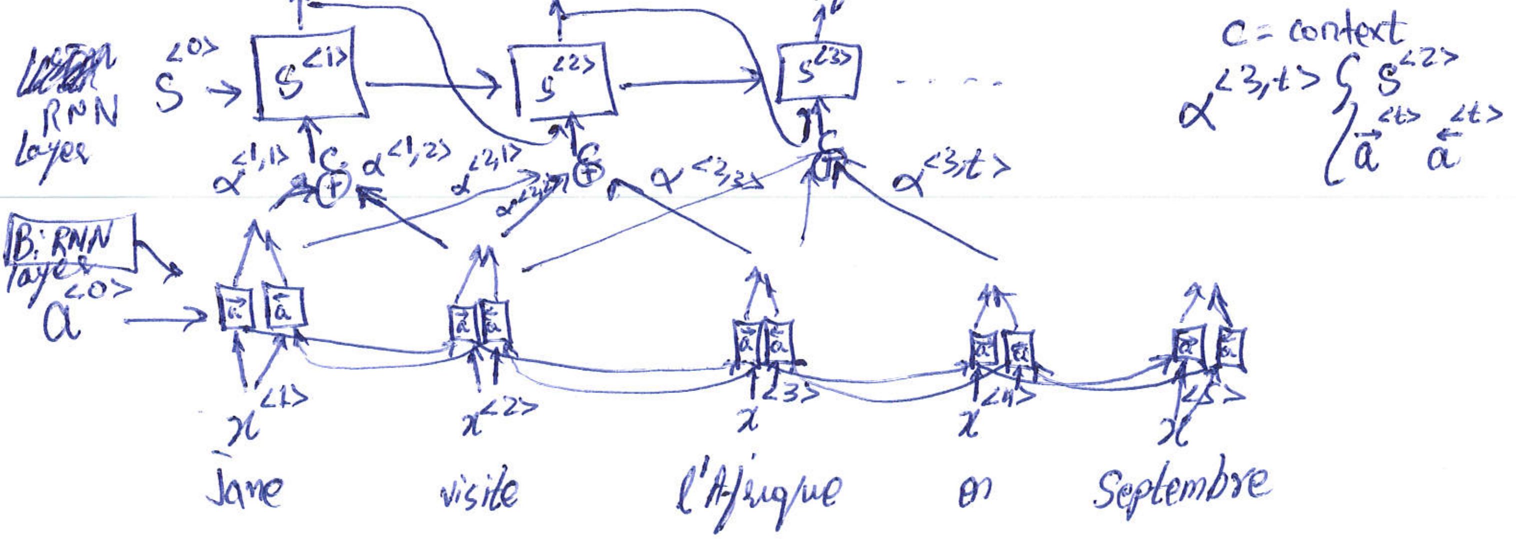
↓ translate to english

Eng Jane went to Africa last .... too.

To translate sentence, we take word & its neighbors translate it to target word rather than reading all text & then translate it to english

PROBLEM





here: $a^{<t'>} = (\overrightarrow{a}^{<t'>}, \overleftarrow{a}^{<t'>})$ Activation features from BiRNN at timestep t'

The context "$C^{<t>}$" is the weighted sum of activation features at time step(t') ~~at~~ or $\overset{\circ}{\circ}'$

$$\Rightarrow C^{<1>} = \sum_{t'} \alpha^{<1,t'>} a^{<t'>}$$

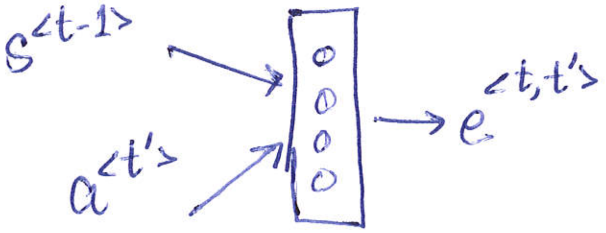
where $a^{<t'>} = (\overrightarrow{a}^{<t'>}, \overleftarrow{a}^{<t'>})$

And $\sum_{t'} \alpha^{<1,t'>} = 1$

$\alpha^{<t,t'>} =$ amount of "attention" $y^{<t>}$ should pay to $a^{<t'>}$

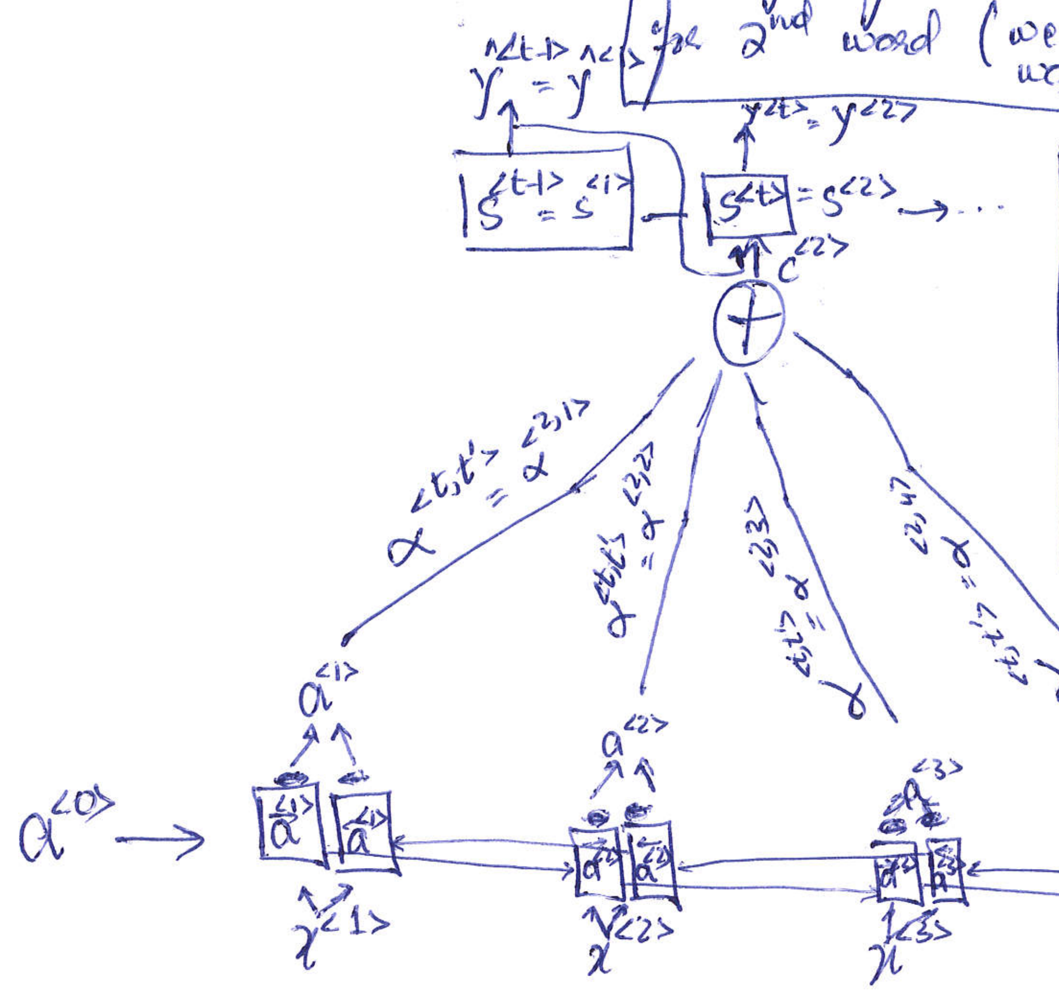
$$\alpha^{<t,t'>} = \frac{\exp(e^{<t,t'>})}{\sum_{t'=1}^{T_x} \exp(e^{<t,t'>})}$$

How to get $e^{<t,t'>}$? One way is to ~~get~~ train small NN & learn a function using FF network & believe in gradient descent



15-C

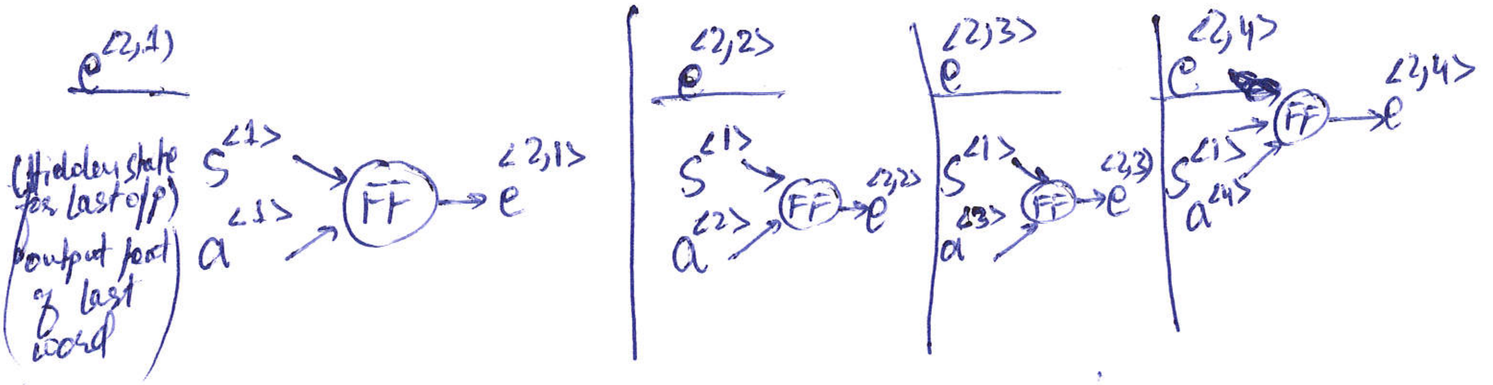
For $y^{<t>} = y^{<2>}$ i.e for 2nd word (we generate output word by word)



$\hat{y}^{<t-1>} = \hat{y}^{<1>}$
$\hat{y}^{<t>} = \hat{y}^{<2>}$
$S^{<t-1>} = S^{<1>}$
$S^{<t>} = S^{<2>}$
$C^{<2>}$
$\alpha^{<t,t'>} = \alpha^{<2,1>}$
$\alpha^{<t,t'>} = \alpha^{<2,2>}$
$\alpha^{<t,t'>} = \alpha^{<2,3>}$
$\alpha^{<t,t'>} = \alpha^{<2,4>}$
$a^{<0>}$
$a^{<1>}$
$a^{<2>}$
$a^{<3>}$
$a^{<4>}$
$x^{<1>}$
$x^{<2>}$
$x^{<3>}$
$x^{<4>}$

Here we generate translation word by word although there are encoder-decoder models for translation

$$\Rightarrow \alpha^{<2,1>} = \frac{\exp\left(e^{<2,1>}\right)}{\exp\left(e^{<2,1>}\right) + \exp\left(e^{<2,2>}\right) + \exp\left(e^{<2,3>}\right) + \exp\left(e^{<2,4>}\right)}$$

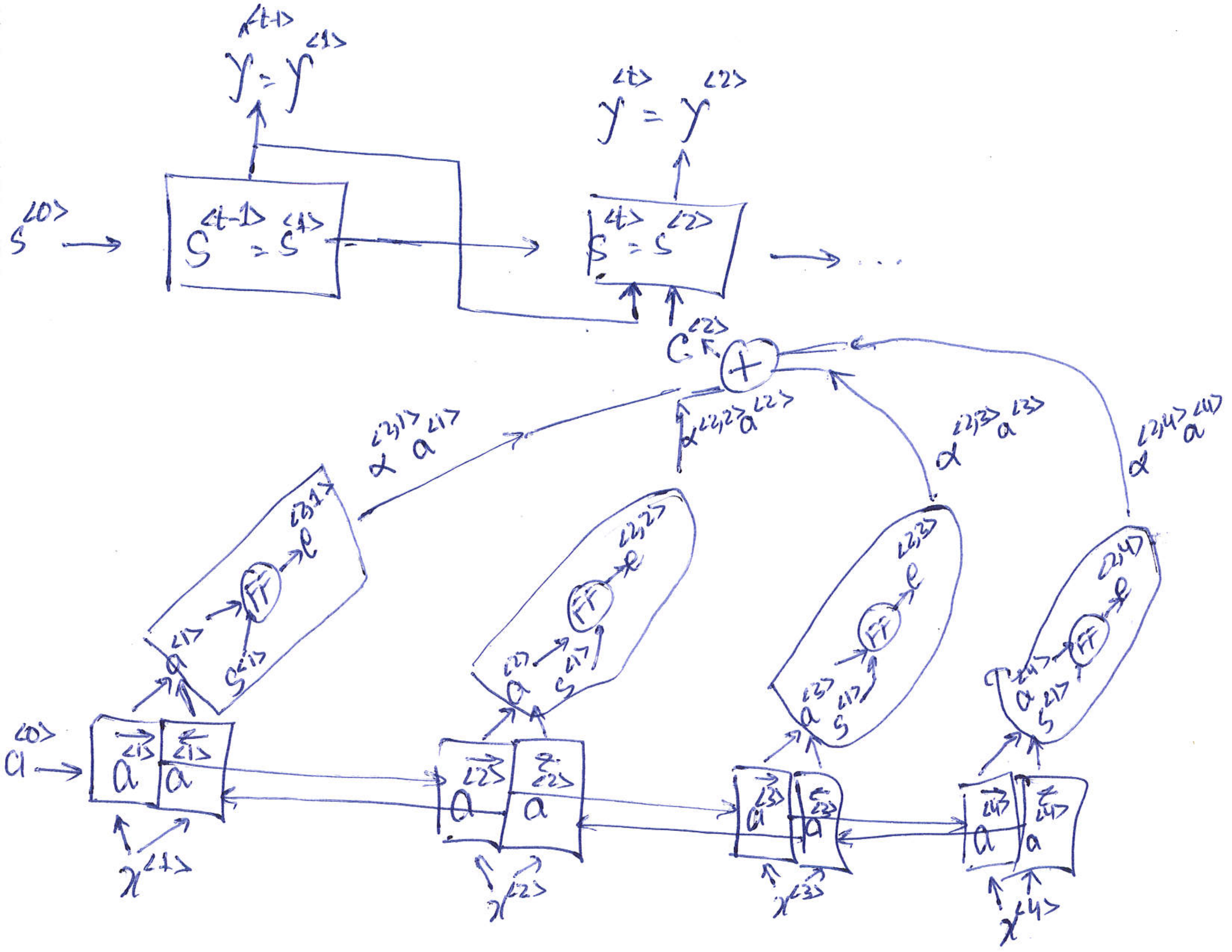


The only downside of this algorithm is it runs in quadratic time i.e if input is T_x & o/p sequence is T_y then number of attention terms will be $(T_x)(T_y)$ & we will run FF $(T_x)(T_y)$ times

And $C^{<2>} = \alpha^{<2,1>} a^{<1>} + \alpha^{<2,2>} a^{<2>} + \alpha^{<2,3>} a^{<3>} + \alpha^{<2,4>} a^{<4>}$

For $y^{<t>} = y^{<2>}$

45-D



$y^{<t-1>} = y^{<1>}$
$y^{<t>} = y^{<2>}$
$s^{<0>}$
$s^{<t-1>} = s^{<1>}$
$s^{<t>} = s^{<2>}$
...
$c^{<2>}$
+
$\alpha^{<2,1>} a^{<1>}$
$\alpha^{<2,2>} a^{<2>}$
$\alpha^{<2,3>} a^{<3>}$
$\alpha^{<2,4>} a^{<4>}$
$e^{<2,1>}$
$e^{<2,2>}$
$e^{<2,3>}$
$e^{<2,4>}$
FF
FF
FF
FF
$a^{<1>}$
$s^{<1>}$
$a^{<2>}$
$s^{<1>}$
$a^{<3>}$
$s^{<1>}$
$a^{<4>}$
$s^{<1>}$
$a^{<0>}$
$\overrightarrow{a}^{<1>}$
$\overleftarrow{a}^{<1>}$
$\overrightarrow{a}^{<2>}$
$\overleftarrow{a}^{<2>}$
$\overrightarrow{a}^{<3>}$
$\overleftarrow{a}^{<3>}$
$\overrightarrow{a}^{<4>}$
$\overleftarrow{a}^{<4>}$
$x^{<1>}$
$x^{<2>}$
$x^{<3>}$
$x^{<4>}$

Kesas Implementation

$a \in R^{(m, Tx, 2 \times n\_a)}$

$S^{t-1} \in R^{(m, n\_s)}$

W3-L9 Speech recognition

Commercial Products trained on > 100,000 hours

- CTC cost for speech recognition
  (connectionist temporal classification)

Graves et al. 2006. Connectionist Temporal Classification: Labelling unsegmented sequence data with RNN

W3-L10 Trigger word detection

Shaukat
Abidi