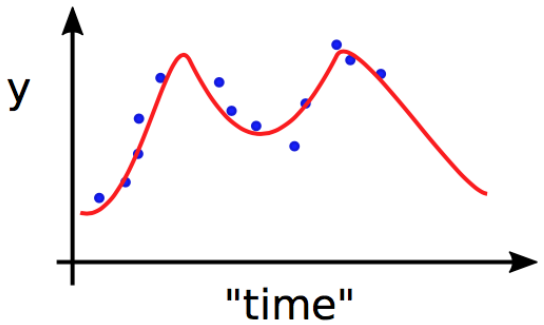
UNIVERSITY OF CAMBRIDGE

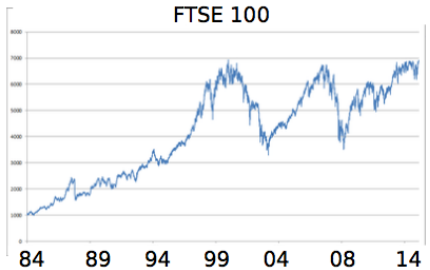
4F10: Deep Learning for Sequence Data

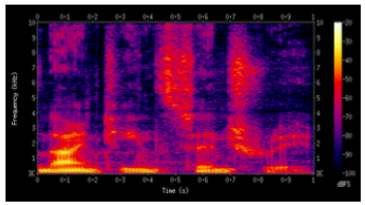
Mark Gales

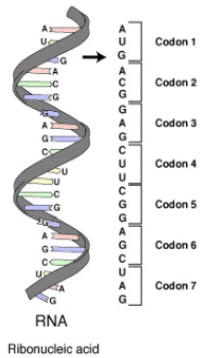
Michaelmas 2020

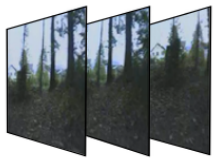
Sequence Modelling (3F8)



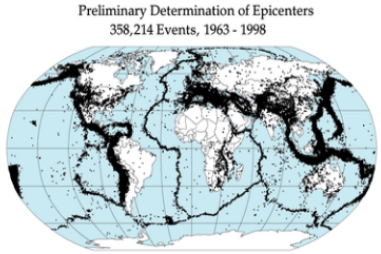








Good King Wenceslas looked out,
On the feast of Stephen,
When the snow lay round about,
Deep and crisp and even;
Brightly shone the moon that night,
Though the frost was cruel,
When a poor man came in sight,
Gathering winter fuel.



I believe that at the end of the century the use of words and general educated opinion will have altered so much that one will be able to speak of machines thinking without expecting to be contradicted. A. Turing

Introduction

- The (input) data considered has been a fixed size
  - vector (iris data), or matrix (image data)

What about variable length data?

- Various sources result in sequences:
  - speech recognition (length of waveform)
  - language processing (length of word sequence)
  - DNA sequences
- Already seen various sequence models (3F8/4F10)
  - hidden Markov model: $p(\boldsymbol{x}_{1:T})$
  - Kalman filter/smoother: $p(\boldsymbol{x}_{1:T})$
  - conditional random field: $P(\boldsymbol{y}_{1:T}|\boldsymbol{x}_{1:T})$

How can deep learning be applied for these tasks

Sequence Modelling Tasks

- Multiple forms of sequence modelling can be considered
  1. sequence input/output pairs: $\{\{\boldsymbol{x}_1, \boldsymbol{y}_1\}, \ldots, \{\boldsymbol{x}_T, \boldsymbol{y}_T\}\}$
  2. input sequence to single output: $\{\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}, \boldsymbol{y}\}$
  3. input sequence to output seq.: $\{\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}, \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_L\}\}$
  4. single input to ouput sequence: $\{\boldsymbol{x}, \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_L\}\}$
  5. sequence distribution modelling: $p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T)$

- T can vary from training sample to training (test) sample
  - for sequence-to-sequence N can vary from sample to sample
- To simplify notation, sequences may be written as

$$\boldsymbol{x}_{1:T} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$$

Sequence Input/Output

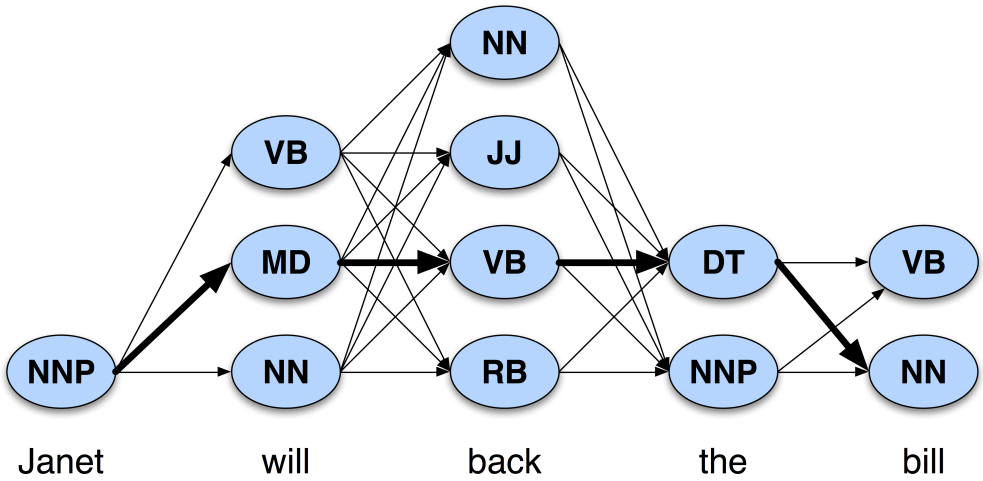
Pair Modelling

Part of Speech Tagging

- Given a word-sequence, $w_{1:L}$, extract PoS tag sequence $y_{1:L}$

  Janet/NNP will/MD back/VB the/DT bill/NN

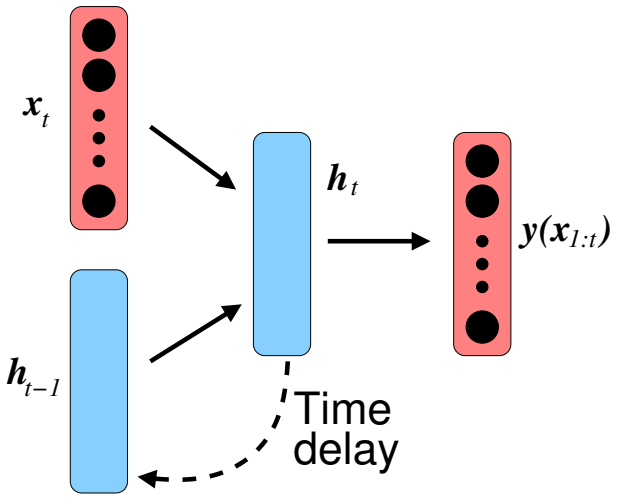
  - individual words are ambiguous (from Jurafsky & Martin)



- Tag sequence (usually) unambiguous given word sequence

Recurrent Neural Networks [17, 16]

- Consider a causal sequence of observations $\boldsymbol{x}_{1:t} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t\}$



  - Introduce recurrent units

$$\boldsymbol{h}_t = \mathbf{f}^{\mathrm{h}}\left(\mathbf{W}^{\mathrm{f}}_{\mathrm{h}}\boldsymbol{x}_t + \mathbf{W}^{\mathrm{r}}_{\mathrm{h}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\mathrm{h}}\right)$$

$$\boldsymbol{y}(\boldsymbol{x}_{1:t}) = \mathbf{f}^{\mathrm{f}}\left(\mathbf{W}_{\mathrm{y}}\boldsymbol{h}_t + \boldsymbol{b}_{\mathrm{y}}\right)$$

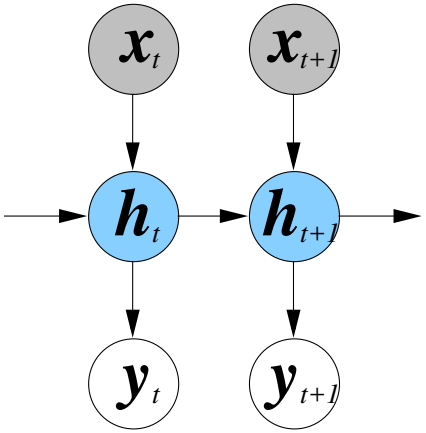
    - $\boldsymbol{h}_t$ history vector at time t
  - Two history weight matrices
    - $\mathbf{W}^{\mathrm{f}}_{\mathrm{h}}$ forward, $\mathbf{W}^{\mathrm{r}}_{\mathrm{h}}$ recursion

- Uses approximation to model history of observations

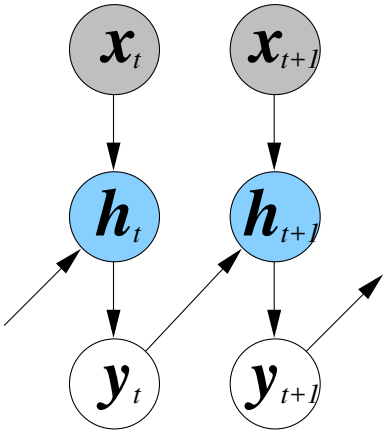
$$\mathcal{F}(\boldsymbol{x}_{1:t}) = \mathcal{F}(\boldsymbol{x}_t, \boldsymbol{x}_{1:t-1}) \approx \mathcal{F}(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}) \approx \mathcal{F}(\boldsymbol{h}_t) = \boldsymbol{y}(\boldsymbol{x}_{1:t}) = \boldsymbol{y}_t$$

  - network has (causal) memory encoded in history vector ($\boldsymbol{h}_t$)

Elman and Jordan Networks



Elman Network

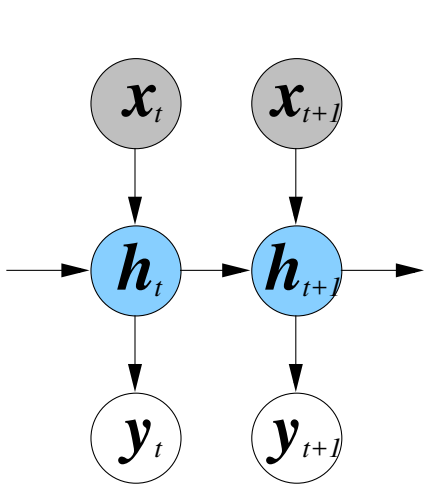


Jordan Network

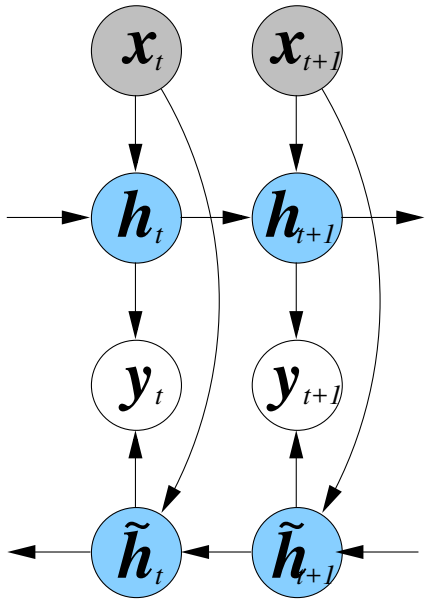
- Previous slide sometimes refered to as an Elman Network
  - history vector, $\boldsymbol{h}_t$ "remembers" input sequence $\boldsymbol{x}_{1:t}$
- Alternative topology, Jordan Network, sometimes used
  - history vector, $\boldsymbol{h}_t$ "remembers" (unobserved) output seq. $\boldsymbol{y}_{1:t}$

$$\mathcal{F}(\boldsymbol{x}_{1:t}, \boldsymbol{y}_{1:t-1}) \approx \mathcal{F}(\boldsymbol{x}_t, \boldsymbol{y}_{1:t-1}) \approx \mathcal{F}(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}) \approx \mathcal{F}(\boldsymbol{h}_t) = \boldsymbol{y}_t$$

RNN Variant: Bi-Directional RNN [19]



RNN

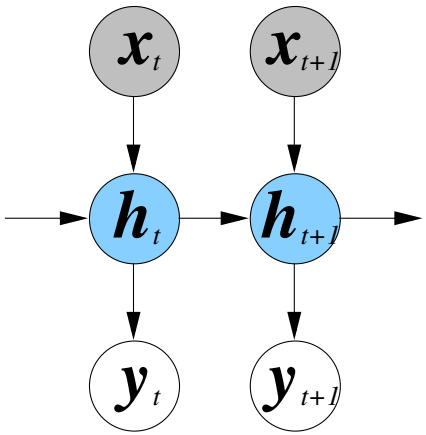


Bi-Directional RNN

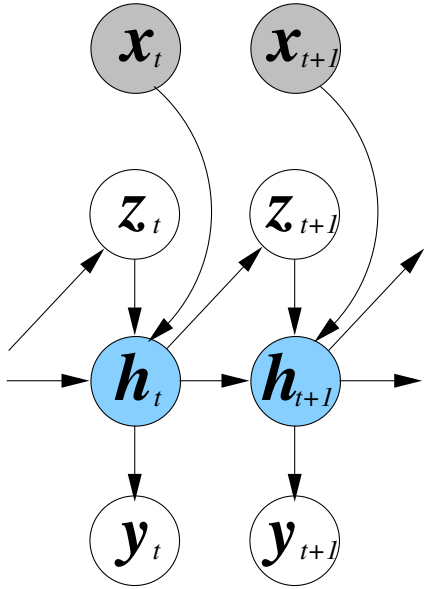
- Bi-directional: use complete observation sequence - non-causal

$$\mathcal{F}_t(\boldsymbol{x}_{1:T}) = \mathcal{F}(\boldsymbol{x}_{1:t}, \boldsymbol{x}_{t:T}) \approx \mathcal{F}(\boldsymbol{h}_t, \tilde{\boldsymbol{h}}_t) = \boldsymbol{y}_t(\boldsymbol{x}_{1:T})$$

Latent Variable (Variational) RNN (reference) [6]



RNN



Variational RNN

- Variational: introduce latent variable sequence $\boldsymbol{z}_{1:T}$

$$p(\boldsymbol{y}_t|\boldsymbol{x}_{1:t}) \approx \int p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{z}_t, \boldsymbol{h}_{t-1}) p(\boldsymbol{z}_t|\boldsymbol{h}_{t-1}) d\boldsymbol{z}_t$$

$$\approx \int p(\boldsymbol{y}_t|\boldsymbol{h}_t) p(\boldsymbol{z}_t|\boldsymbol{h}_{t-1}) d\boldsymbol{z}_t$$

  - $\boldsymbol{z}_t$ a function of complete history (complicates training)

Network Gating

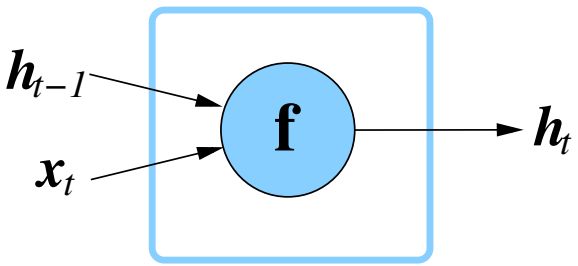
- A flexible extension to activation function is gating
  - standard form is ($\sigma()$ sigmoid activation function)

$$\boldsymbol{i} = \sigma(\mathbf{W}^{\mathrm{f}}\boldsymbol{x}_t + \mathbf{W}^{\mathrm{r}}\boldsymbol{h}_{t-1} + \boldsymbol{b})$$

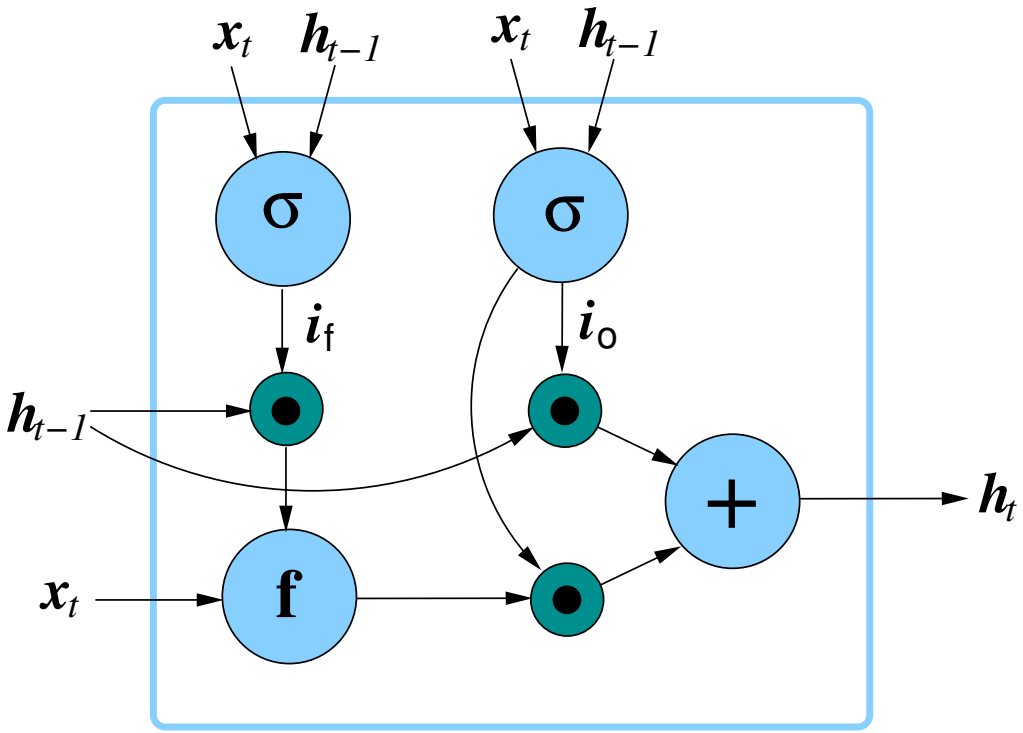
  - vector acts a probabilistic gate on network values

- Gating can be applied at various levels
  - features: impact of input/output features on nodes
  - time: memory of the network
  - layer: influence of a layer's activation function

Gated Recurrent Unit [5]



Recurrent unit



Gated Recurrent Unit

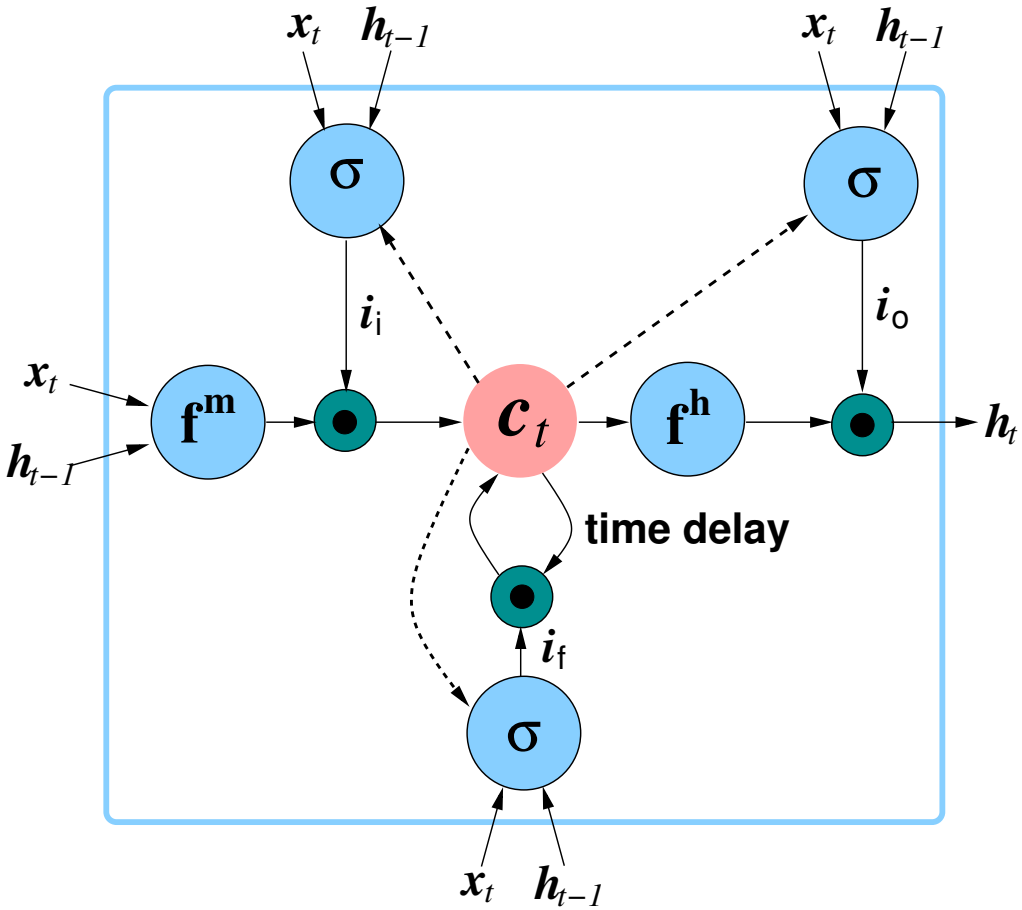
Gated Recurrent Unit [5]

- Gated Recurrent Unit (GRU) introduces
  - forget gate ($\boldsymbol{i}_{\mathtt{f}}$): gating over time
  - output gate ($\boldsymbol{i}_{\mathtt{o}}$): gating over features (and time)
- Relationships (standard configuration - there are variants)

$$
\begin{aligned}
\boldsymbol{i}_{\mathtt{f}} &= \sigma(\mathbf{W}_{\mathtt{f}}^{\mathtt{f}}\boldsymbol{x}_t + \mathbf{W}_{\mathtt{f}}^{\mathtt{r}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\mathtt{f}}) \\
\boldsymbol{i}_{\mathtt{o}} &= \sigma(\mathbf{W}_{\mathtt{o}}^{\mathtt{f}}\boldsymbol{x}_t + \mathbf{W}_{\mathtt{o}}^{\mathtt{r}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\mathtt{o}}) \\
\tilde{\boldsymbol{h}}_t &= \mathbf{f}(\mathbf{W}_{\mathtt{h}}^{\mathtt{f}}\boldsymbol{x}_t + \mathbf{W}_{\mathtt{h}}^{\mathtt{r}}(\boldsymbol{i}_{\mathtt{f}} \odot \boldsymbol{h}_{t-1}) + \boldsymbol{b}_{\mathtt{h}}) \\
\boldsymbol{h}_t &= \boldsymbol{i}_{\mathtt{o}} \odot \boldsymbol{h}_{t-1} + (\mathbf{1} - \boldsymbol{i}_{\mathtt{o}}) \odot \tilde{\boldsymbol{h}}_t
\end{aligned}
$$

  - $\odot$ represents element-wise multiplication between vectors

Long-Short Term Memory Networks [11, 9]



Long-Short Term Memory Networks

- The operations can be written as (peephole config):
  - Forget gate ($\boldsymbol{i}_{\mathtt{f}}$), Input gate ($\boldsymbol{i}_{\mathtt{i}}$), Output gate ($\boldsymbol{i}_{\mathtt{o}}$)

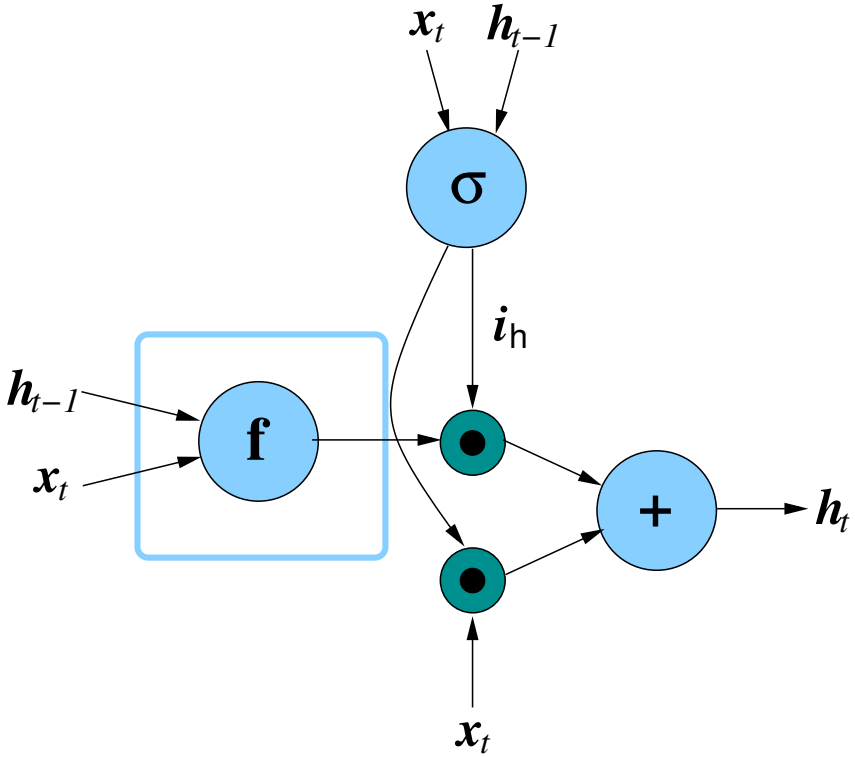
$$
\begin{aligned}
\boldsymbol{i}_{\mathtt{f}} &= \sigma(\mathbf{W}^{\mathtt{f}}_{\mathtt{f}}\boldsymbol{x}_t + \mathbf{W}^{\mathtt{r}}_{\mathtt{f}}\boldsymbol{h}_{t-1} + \mathbf{W}^{\mathtt{m}}_{\mathtt{f}}\boldsymbol{c}_{t-1} + \boldsymbol{b}_{\mathtt{f}}) \\
\boldsymbol{i}_{\mathtt{i}} &= \sigma(\mathbf{W}^{\mathtt{f}}_{\mathtt{i}}\boldsymbol{x}_t + \mathbf{W}^{\mathtt{r}}_{\mathtt{i}}\boldsymbol{h}_{t-1} + \mathbf{W}^{\mathtt{m}}_{\mathtt{i}}\boldsymbol{c}_{t-1} + \boldsymbol{b}_{\mathtt{i}}) \\
\boldsymbol{i}_{\mathtt{o}} &= \sigma(\mathbf{W}^{\mathtt{f}}_{\mathtt{o}}\boldsymbol{x}_t + \mathbf{W}^{\mathtt{r}}_{\mathtt{o}}\boldsymbol{h}_{t-1} + \mathbf{W}^{\mathtt{m}}_{\mathtt{o}}\boldsymbol{c}_{t} + \boldsymbol{b}_{\mathtt{o}})
\end{aligned}
$$

  - Memory Cell, history vector and gates are related by

$$
\begin{aligned}
\boldsymbol{c}_t &= \boldsymbol{i}_{\mathtt{f}} \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_{\mathtt{i}} \odot \mathbf{f}^{\mathtt{m}}(\mathbf{W}^{\mathtt{f}}_{\mathtt{c}}\boldsymbol{x}_t + \mathbf{W}^{\mathtt{r}}_{\mathtt{c}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\mathtt{c}}) \\
\boldsymbol{h}_t &= \boldsymbol{i}_{\mathtt{o}} \odot \mathbf{f}^{\mathtt{h}}(\boldsymbol{c}_t)
\end{aligned}
$$

  - more complicated than GRU (three gates, memory cell)
  - memory cell weight matrices ($\mathbf{W}^{\mathtt{m}}_{\mathtt{f}}, \mathbf{W}^{\mathtt{m}}_{\mathtt{i}}, \mathbf{W}^{\mathtt{m}}_{\mathtt{o}}$) diagonal
  - can allow explicit analysis of individual cell elements

Highway Connections [20]



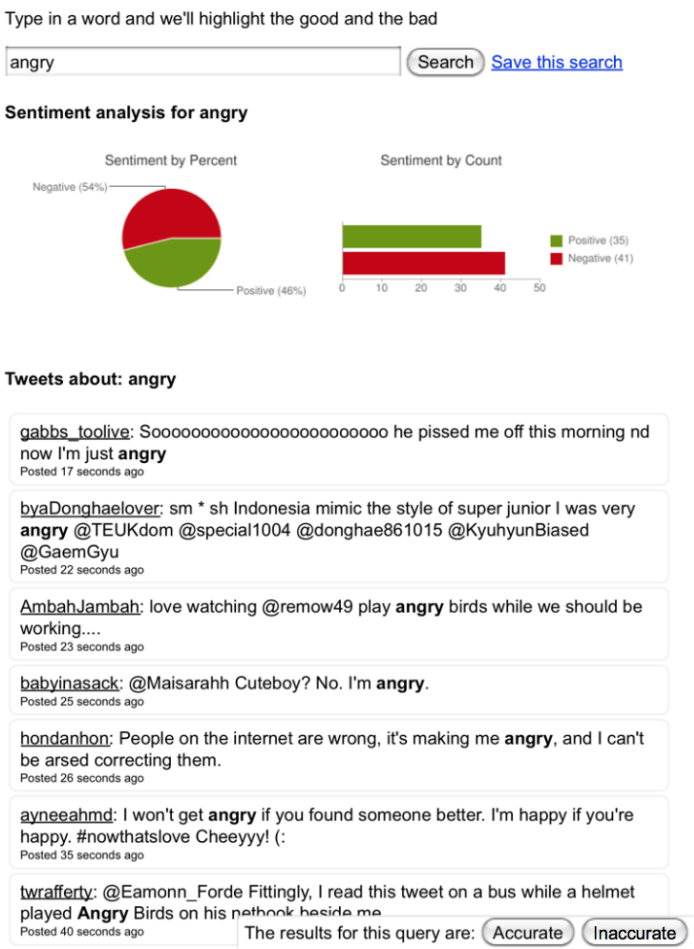
- Gate the output of the node (example from recurrent unit)
  - combine with output from previous layer ($\boldsymbol{x}_t$)

$$\begin{aligned} \boldsymbol{i}_{\mathtt{h}} &= \sigma(\mathbf{W}^{\mathtt{f}}_{\mathtt{l}}\boldsymbol{x}_t + \mathbf{W}^{\mathtt{r}}_{\mathtt{l}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\mathtt{l}}) \\ \boldsymbol{h}_t &= \boldsymbol{i}_{\mathtt{h}} \odot \mathbf{f}(\mathbf{W}^{\mathtt{f}}_{\mathtt{h}}\boldsymbol{x}_t + \mathbf{W}^{\mathtt{r}}_{\mathtt{h}}\boldsymbol{h}_{t-1} + \boldsymbol{b}_{\mathtt{h}}) + (\mathbf{1} - \boldsymbol{i}_{\mathtt{h}}) \odot \boldsymbol{x}_t \end{aligned}$$

Input Sequence to Target

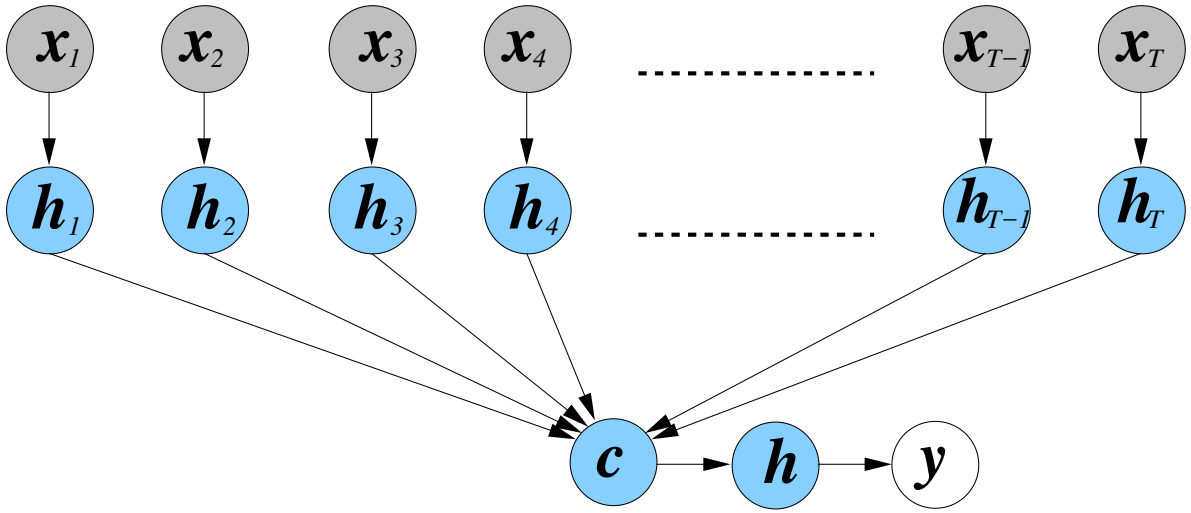
Twitter Sentiment Analysis

- Given word sequence, $w_{1:L}$, extract sentiment $\{\texttt{pos}, \texttt{neg}\}$



- "Twitter Sentiment" for `angry`
  - tweets about `angry birds` game
  - sentiment accuracy reasonable
- Ambiguous for individual words

Averaging



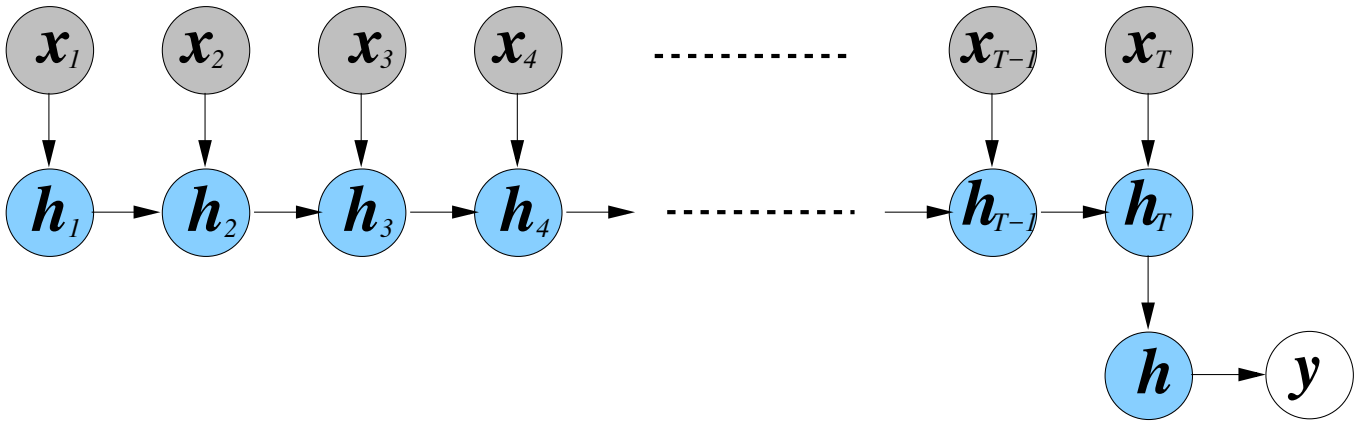
- Consider regression/classification of the form

$$y = \mathcal{F}(x_1, \ldots, x_T)$$

  - need to map T (variable) length sequence to a fixed length
- Network above is one simple form

$$y = \mathcal{F}(h); \quad h = \mathcal{F}(c); \quad c = \frac{1}{T}\sum_{t=1}^{T} h_t; \quad h_t = \mathcal{F}(x_t)$$

RNN Encoding (Sequence Embedding)



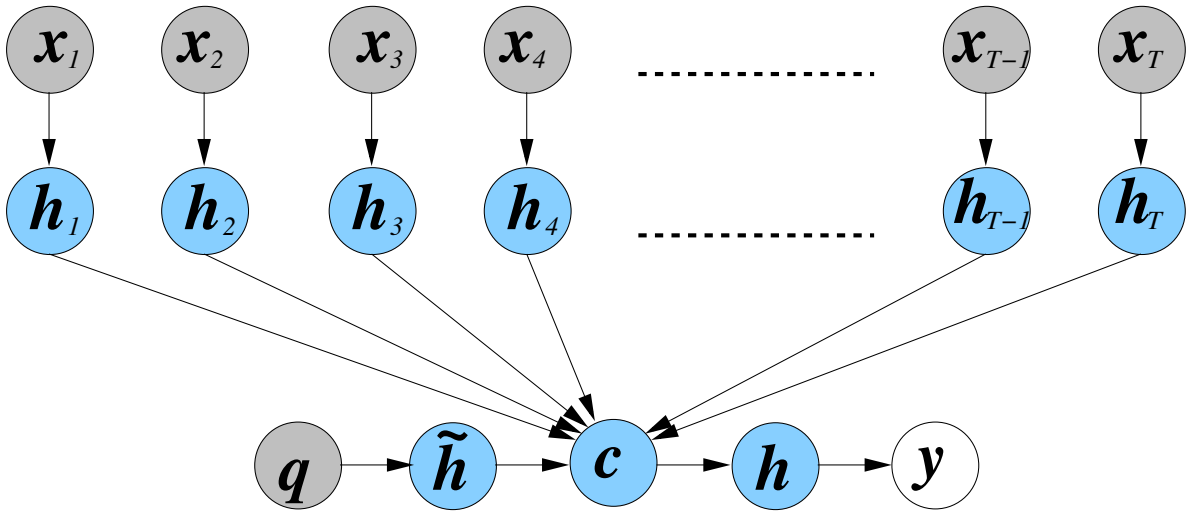
- Alternative use final history vector as information

$$\boldsymbol{h} = \mathcal{F}(\boldsymbol{h}_T); \quad \boldsymbol{h}_t = \mathcal{F}(\boldsymbol{x}_t, \boldsymbol{h}_{t-1})$$

- Handles variable length, but biased to later inputs
  - can use bi-directional information

$$\boldsymbol{h} = \mathcal{F}\left(\boldsymbol{h}_T, \tilde{\boldsymbol{h}}_1\right); \quad \boldsymbol{h}_t = \mathcal{F}(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}); \quad \tilde{\boldsymbol{h}}_t = \mathcal{F}(\boldsymbol{x}_t, \tilde{\boldsymbol{h}}_{t+1})$$

Attention Mechanism



- Extend average to be a function of some query, $\boldsymbol{q}$,

$$\boldsymbol{c} = \sum_{t=1}^{T} \alpha_t \boldsymbol{h}_t; \quad \alpha_t = \frac{\exp(e_t)}{\sum_{i=1}^{T} \exp(e_i)}; \quad e_t = \mathcal{F}(\tilde{\boldsymbol{h}}, \boldsymbol{h}_t); \quad \tilde{\boldsymbol{h}} = \mathcal{F}(\boldsymbol{q})$$

  - yields a probability mass function over the sequence
  - important to get the appropriate form of $\boldsymbol{q}$

Form of Attention Mechanism

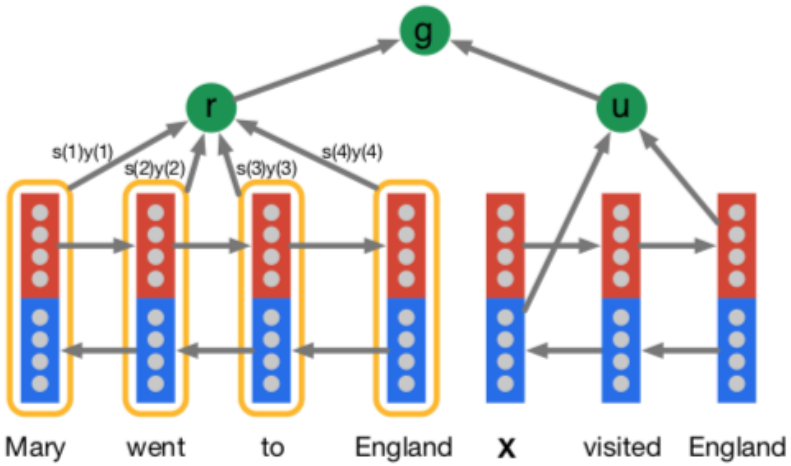
- At the center of the attention mechanism is $e_t = \mathcal{F}(\tilde{\boldsymbol{h}}, \boldsymbol{h}_t)$
  - relates (embedded) query $\boldsymbol{q}$ to (embedded) obeservation $\boldsymbol{x}_t$
  - how relevant is that observation to the query
- Two standard forms of attention are:
  - dot-product attention (parameters $\mathbf{W}_{\mathtt{xq}}$)

$$e_t = \boldsymbol{h}_t^{\mathsf{T}} \mathbf{W}_{\mathtt{xq}} \tilde{\boldsymbol{h}}$$

  - additive attention (parameters $\mathbf{w}, \mathbf{W}_{\mathtt{x}}, \mathbf{W}_{\mathtt{q}}$)

$$e_t = \mathbf{w}^{\mathsf{T}} \tanh\left(\mathbf{W}_{\mathtt{x}} \boldsymbol{h}_t + \mathbf{W}_{\mathtt{q}} \tilde{\boldsymbol{h}}\right)$$

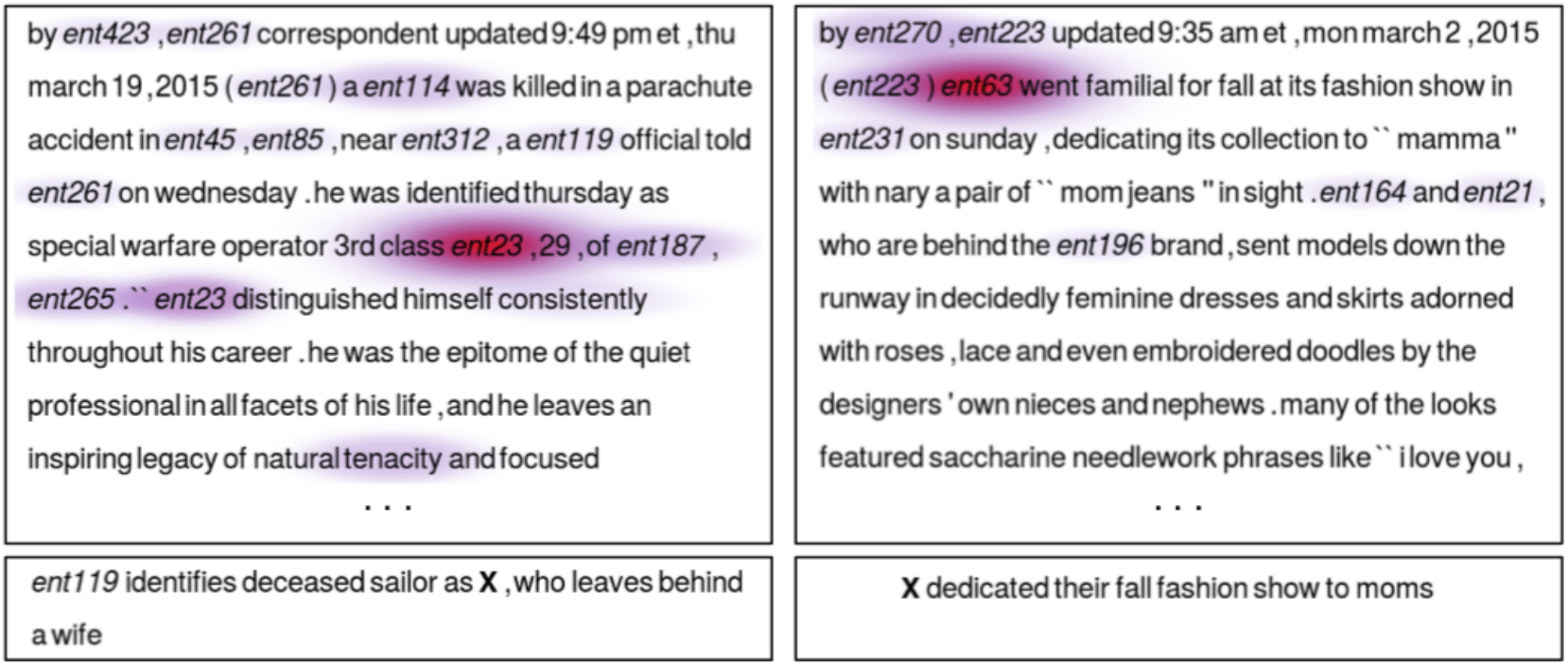
Attentive Reader [10]



- Example question-answering network combines approaches
  - question encoding - forward, $\boldsymbol{u}_{|q|}$, and backwards, $\tilde{\boldsymbol{u}}_1$
  - document encoding - bi-directional encoding per word, $\boldsymbol{y}_t, \tilde{\boldsymbol{y}}_t$ followed by attention mechanism

$$\boldsymbol{r} = \sum_{t=1}^{|d|} s_t \begin{bmatrix} \boldsymbol{y}_t \\ \tilde{\boldsymbol{y}}_t \end{bmatrix}; \quad s_t = \frac{1}{Z} \exp\left(\mathbf{w}^\top \tanh\left(\mathbf{W}_{\text{ym}} \begin{bmatrix} \boldsymbol{y}_t \\ \tilde{\boldsymbol{y}}_t \end{bmatrix} + \mathbf{W}_{\text{um}} \begin{bmatrix} \boldsymbol{u}_{|q|} \\ \tilde{\boldsymbol{u}}_1 \end{bmatrix}\right)\right)$$

Attentive Reader



- Attention mechanisms can be interpretable
  - can examine which word in document is most relevant

Sequence to Sequence

Machine Translation

UNIVERSITY OF CAMBRIDGE — DEPARTMENT OF ENGINEERING

Marquer les rafales

Les rafales de marque est un lecteur dans la technologie de l'information dans le laboratoire d'intelligence de machine (autrefois le groupe de vision et de robotique de la parole (SVR)) et un camarade de l'université d'Emmanuel. Il est un membre du groupe de recherche de la parole ainsi que les jeunes de Steve de membres de personnel de corps enseignant, la régfion boisée et la facture Byrne de Phil.

Une brève biographie est accessible en ligne.

[Recherche | projets | publications | étudiants | enseignant | contact]

Intérêts de recherches

- Reconnaissance de la parole continue de grand vocabulaire
- Reconnaissance de la parole robuste
- Adaptation d'orateur
- Étude de machine (en particulier choix modèle et méthodes grain-basées)
- Identification et vérification d'orateur

Une brève introduction à la reconnaissance de la parole est accessible en ligne.
dessus

Projets de recherche

Projets en cours :

- Bruit ASR robuste (Europe Ltd de recherches de Toshiba placée)
- Traitement averti d'environnement rapide et robuste (Europe Ltd de recherches de Toshiba placée)
  - NEW Position d'associé de recherches disponible
- AGILE (projet placé par GALE de DARPA)
- Version 3 de HTK - HTK_V3.4 et exemples sont disponibles.

Projets récemment réalisés :

- CoreTex (améliorant la technologie de reconnaissance de la parole de noyau)
- Transcription audio riche de HTK(Projet placé par OREILLES de DARPA) - pages Web locaux

- Google Translate (2009)
  - applied to my web-page
  - target language French
- Ambiguous for individual words
- Interesting output for people
  - Mark Gales
  - Bill Byrne
  - Phil Woodland
  - Steve Young

Encoder-Decoder Sequence Models

- Train a discriminative model from
  - $\boldsymbol{x}_{1:L} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_L\}$: L-length input sequence
  - $\boldsymbol{y}_{1:K} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_K\}$: K-length output

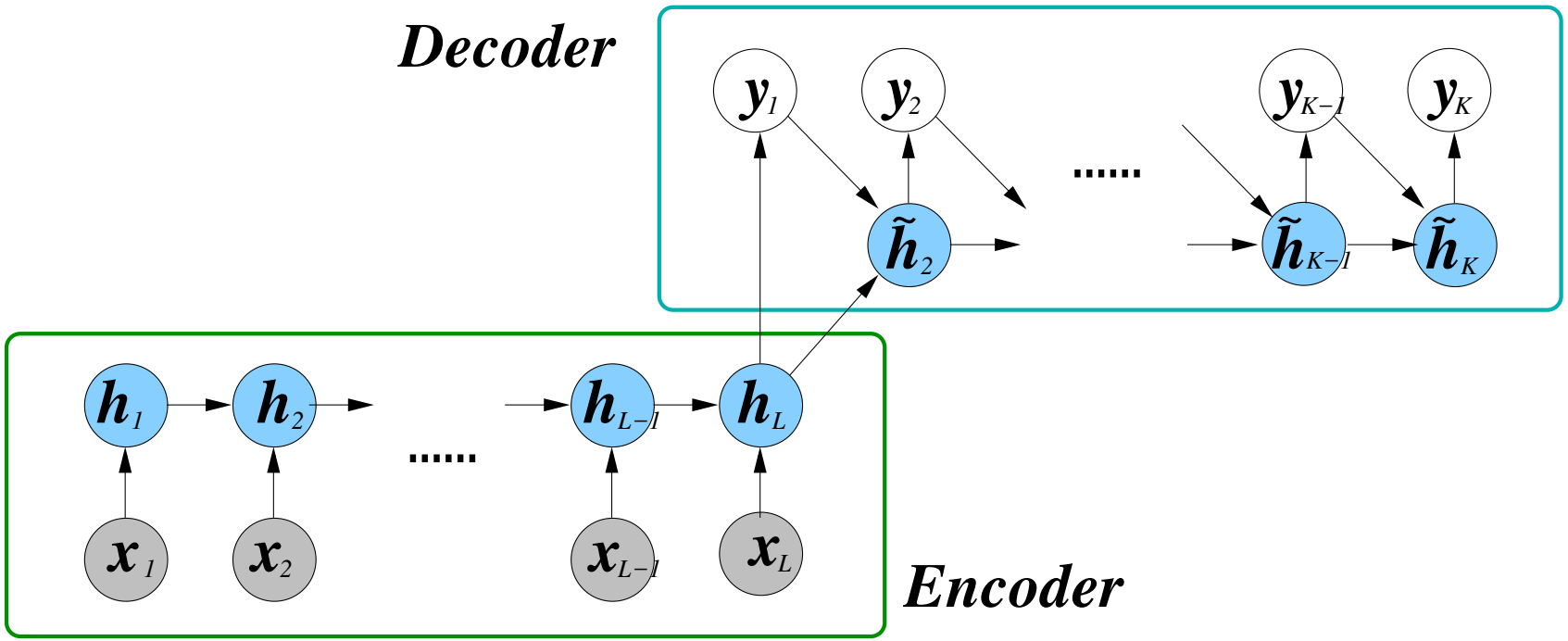
$$
\begin{aligned}
p(\boldsymbol{y}_{1:K}|\boldsymbol{x}_{1:L}) &= \prod_{i=1}^{K} p(\boldsymbol{y}_i|\boldsymbol{y}_{1:i-1}, \boldsymbol{x}_{1:L}) \\
&\approx \prod_{i=1}^{L} p(\boldsymbol{y}_i|\boldsymbol{y}_{i-1}, \tilde{\boldsymbol{h}}_{i-1}, \boldsymbol{c})
\end{aligned}
$$

  - need to map $\boldsymbol{x}_{1:L}$ to a fixed-length vector

$$\boldsymbol{c} = \phi(\boldsymbol{x}_{1:L})$$

  - $\boldsymbol{c}$ is a fixed length vector - like a sequence kernel

RNN Encoder-Decoder Model [8, 14]

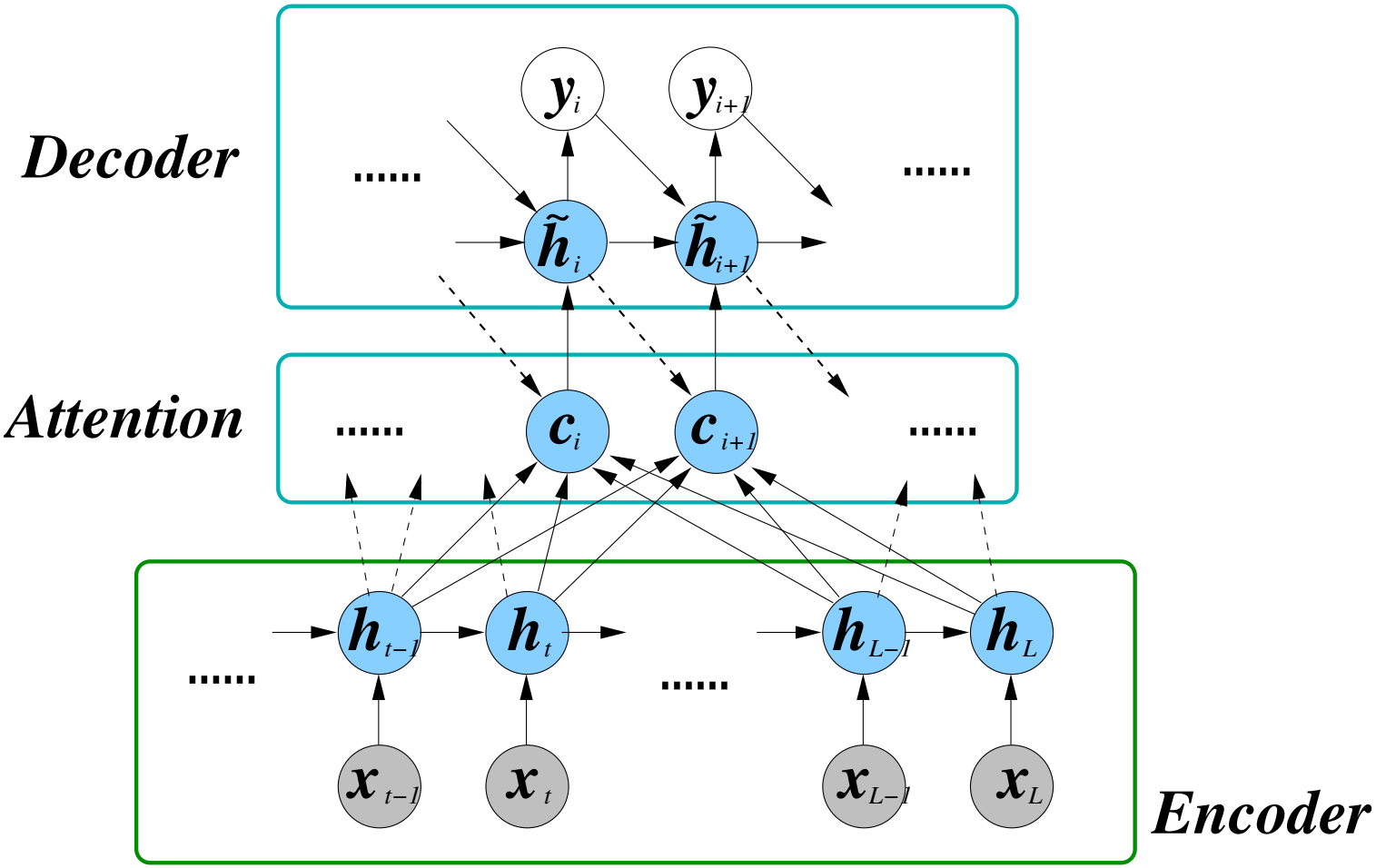


- One form is to use hidden unit from acoustic RNN/LSTM

$$\boldsymbol{c} = \phi(\boldsymbol{x}_{1:L}) = \boldsymbol{h}_L$$

  - dependence on context is global via $\boldsymbol{c}$ - possibly limiting

Attention-Based Models [4, 3, 14]



Attention-Based Models

- Introduce attention layer to system
  - introduce dependence on locality i

$$p(\boldsymbol{y}_{1:K}|\boldsymbol{x}_{1:L}) \approx \prod_{i=1}^{K} p(\boldsymbol{y}_i|\boldsymbol{y}_{i-1}, \tilde{\boldsymbol{h}}_{i-1}, \boldsymbol{c}_i) \approx \prod_{i=1}^{K} p(\boldsymbol{y}_i|\tilde{\boldsymbol{h}}_i)$$

where

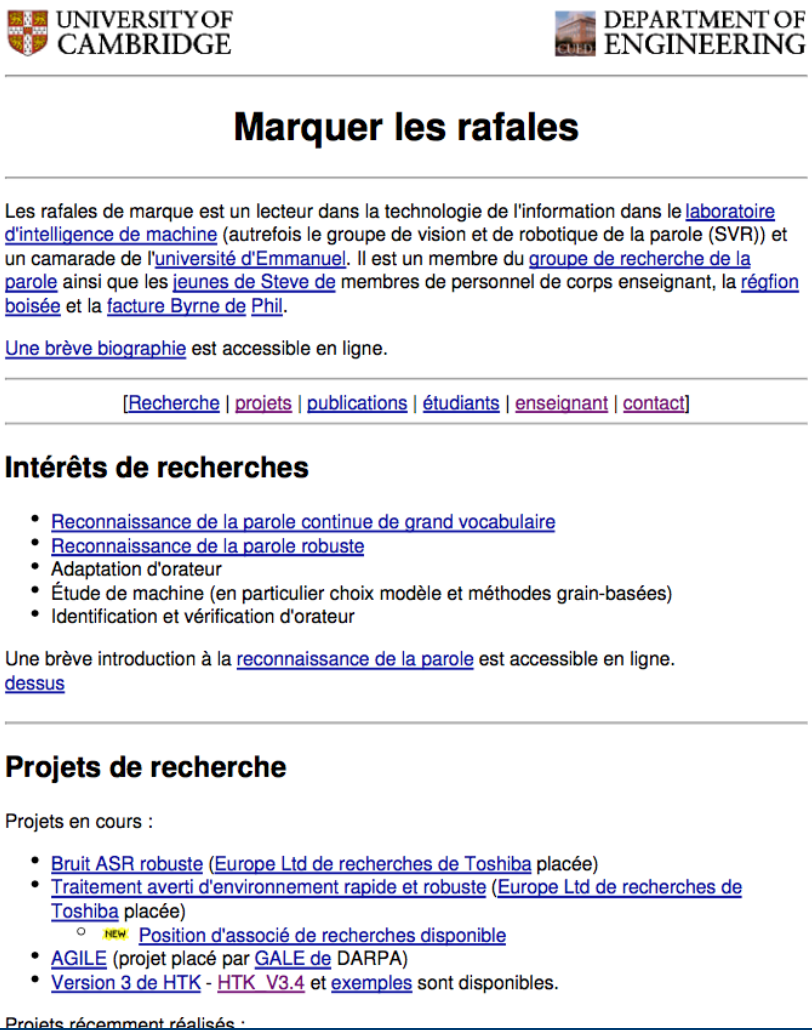
$$\boldsymbol{c}_i = \sum_{\tau=1}^{L} \alpha_{i\tau} \boldsymbol{h}_\tau; \quad \alpha_{i\tau} = \frac{\exp(e_{i\tau})}{\sum_{j=1}^{L} \exp(e_{ij})}, \quad e_{i\tau} = \mathcal{F}\left(\tilde{\boldsymbol{h}}_{i-1}, \boldsymbol{h}_\tau\right)$$

  - $e_{i\tau}$ how well position τ in input predicts position i in output
  - $\boldsymbol{h}_\tau$ is representation (RNN) for the input at position τ

- Uses output history $\boldsymbol{y}_{1:i-1}$ as query for i^{th} word

Machine Translation

- Given word sequence, $\boldsymbol{w}_{1:L}$, translate to French $\hat{\boldsymbol{y}}_{1:K}$



UNIVERSITY OF CAMBRIDGE

DEPARTMENT OF ENGINEERING

Marquer les rafales

Les rafales de marque est un lecteur dans la technologie de l'information dans le laboratoire d'intelligence de machine (autrefois le groupe de vision et de robotique de la parole (SVR)) et un camarade de l'université d'Emmanuel. Il est un membre du groupe de recherche de la parole ainsi que les jeunes de Steve de membres de personnel de corps enseignant, la régfion boisée et la facture Byrne de Phil.

Une brève biographie est accessible en ligne.

[Recherche | projets | publications | étudiants | enseignant | contact]

Intérêts de recherches

- Reconnaissance de la parole continue de grand vocabulaire
- Reconnaissance de la parole robuste
- Adaptation d'orateur
- Étude de machine (en particulier choix modèle et méthodes grain-basées)
- Identification et vérification d'orateur

Une brève introduction à la reconnaissance de la parole est accessible en ligne.
dessus

Projets de recherche

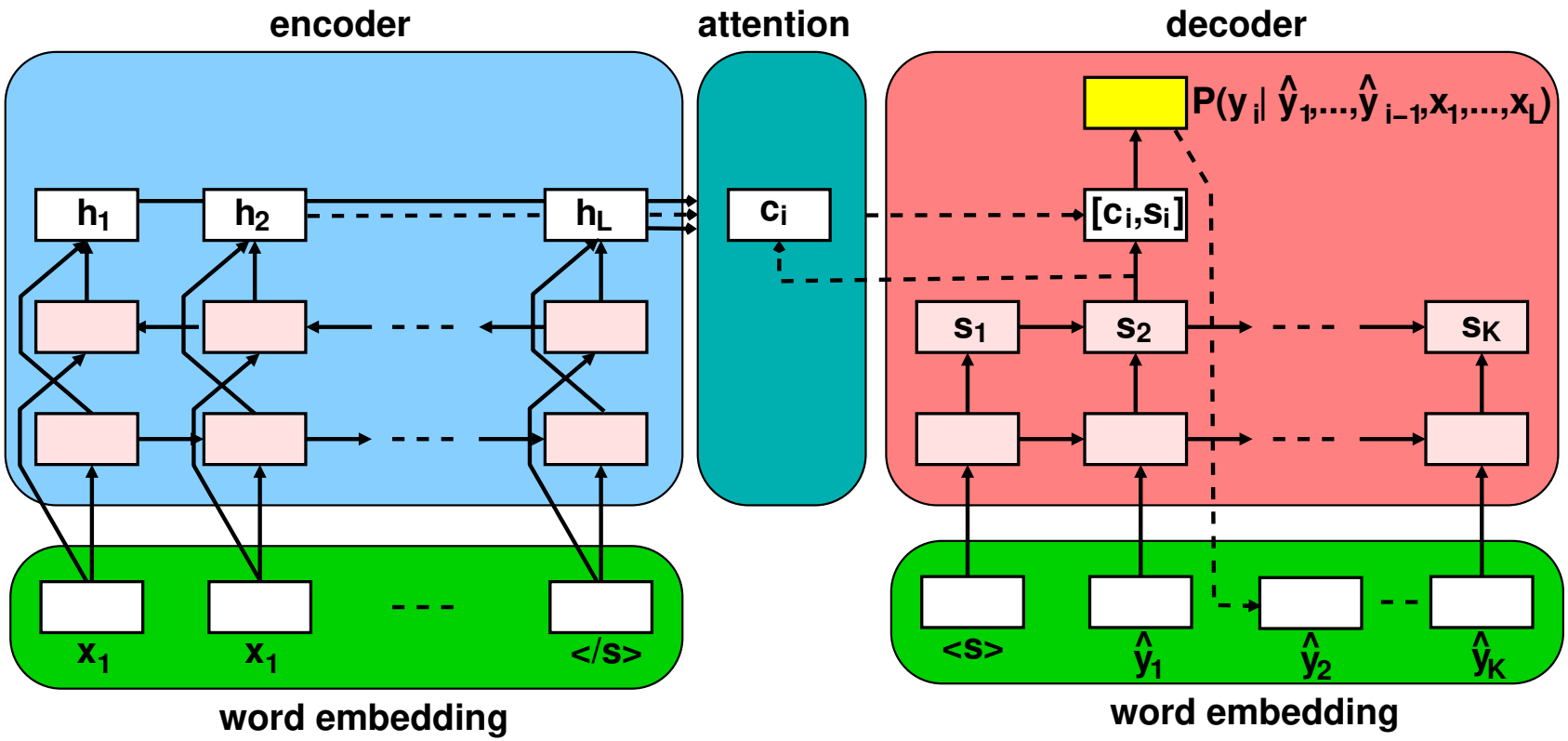
Projets en cours :

- Bruit ASR robuste (Europe Ltd de recherches de Toshiba placée)
- Traitement averti d'environnement rapide et robuste (Europe Ltd de recherches de Toshiba placée)
  - NEW Position d'associé de recherches disponible
- AGILE (projet placé par GALE de DARPA)
- Version 3 de HTK - HTK_V3.4 et exemples sont disponibles.

Projets récemment réalisés :

- Google Translate (2009)
  - applied to my web-page
  - target language French
- Ambiguous for individual words
- Interesting output for people
  - Mark Gales
  - Bill Byrne
  - Phil Woodland
  - Steve Young

(Simplified) Neural Machine Translation



- Encoder-decoder framework with attention
  - word-embedding maps word (`cat`) to vector (`emb(cat)`)
- Current state-of-the-art more complicated see MT papers

Neural Machine Translation - Distribution

- Overall probability distribution over all outputs based on

$$\hat{\boldsymbol{y}}_{1:K} \sim P(\boldsymbol{y}_{1:K}|\boldsymbol{w}_{1:L}) = P(\boldsymbol{y}_{1:K}|\boldsymbol{x}_{1:L})$$

  - $\boldsymbol{x}_{1:L}$ sequence of word embeddings for word sequence $\boldsymbol{w}_{1:L}$
  - generate translated word sequence $\hat{\boldsymbol{y}}_{1:K}$ from this distribution (swapped to $P()$ as the set of possible translations is discrete)

- Similar to language modelling use conditional distributions

$$P(\boldsymbol{y}_{1:K}|\boldsymbol{x}_{1:L}) = \prod_{i=1}^{K} P(y_i|\boldsymbol{y}_{1:i-1}, \boldsymbol{x}_{1:L})$$

  - need: a representation of: input embedding sequence $\boldsymbol{x}_{1:L}$; (conditional) representation of the previous output words $\boldsymbol{y}_{1:i-1}$

Neural Machine Translation - Attention

- Possible translation generated in an autoregressive fashion

$$\hat{y}_i \sim P(y_i|\hat{\boldsymbol{y}}_{1:i-1}, \boldsymbol{x}_{1:L}) \approx P(y_i|\boldsymbol{s}_i, \boldsymbol{c}_i)$$

where

$$\hat{y}_i \sim \mathcal{F}_{\texttt{softmax}}(\boldsymbol{s}_i, \boldsymbol{c}_i); \quad \boldsymbol{s}_i = \mathcal{F}_{\texttt{rnn}}(\boldsymbol{s}_{i-1}, \tilde{\boldsymbol{h}}_i); \quad \tilde{\boldsymbol{h}}_i = \mathcal{F}_{\texttt{rnn}}(\tilde{\boldsymbol{h}}_{i-1}, \texttt{emb}(\hat{y}_{i-1}))$$

$$\boldsymbol{c}_i = \mathcal{F}_{\texttt{att}}(\boldsymbol{s}_i, \boldsymbol{h}_{1:L}) = \sum_{\tau=1}^{L} \alpha_{i\tau} \boldsymbol{h}_\tau; \quad \alpha_{i\tau} = \frac{\exp(e_{i\tau})}{\sum_{j=1}^{L} \exp(e_{ij})}, \quad e_{i\tau} = \mathcal{F}(\boldsymbol{s}_i, \boldsymbol{h}_\tau)$$

  - $e_{i\tau}$ how well position τ in input predicts position i in output
  - $\boldsymbol{h}_\tau$ is representation (bi-RNN) for the input at position τ

- At inference time need to find most probable translations
  - simplest approach is greedy search

Greedy Inference for NMT

- The decoder part of the model related to Jordan network
  - history vector encodes outputs and context histories

$$\hat{y}_i = \arg\max_{\omega} \{P(\omega|\boldsymbol{s}_i, \boldsymbol{c}_i)\}$$

  - where $\omega \in \mathcal{V}$, $\mathcal{V}$ is the NMT vocabulary
- Sequence generation process:
  1. generate word-embeddings for source sentence $\boldsymbol{x}_{1:L}$
  2. generate encoding of source sentence $\boldsymbol{h}_{1:L} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_L\}$
  3. initialise decoding: $\hat{y}_0 = <\mathrm{s}>$, $\boldsymbol{s}_0$, $\tilde{\boldsymbol{h}}_0$, set $i = 0$
  4. DECODE: until $\hat{y}_i = </\mathrm{s}>$ do
     - 4.1 $i = i + 1$
     - 4.2 compute new "state" vector $\boldsymbol{s}_i$
     - 4.3 compute "context" information $\boldsymbol{c}_i = \mathcal{F}_{\mathrm{att}}(\boldsymbol{s}_i, \boldsymbol{h}_{1:L})$
     - 4.4 obtain distribution over target language words $P(y_i|\boldsymbol{s}_i, \boldsymbol{c}_i)$
     - 4.5 sample from distribution $\hat{y}_i$, append to sentence $\rightarrow \hat{y}_{1:i}$

Teacher-Forcing (reference) [1, 12]

- One issue is that errors can occur in the history $\hat{\boldsymbol{y}}_{0:i-1}$
  - system usually trained on correct data (bitext for MT)
    this is sometimes refered to as teacher-forcing
- But at inference (decoding) time system can go "off-track"
  - would like to make system robust to model history errors
- Approaches have been developed to address this problem
  - scheduled sampling: mix "correct" history with model history
  - professor forcing: related to adversarial training

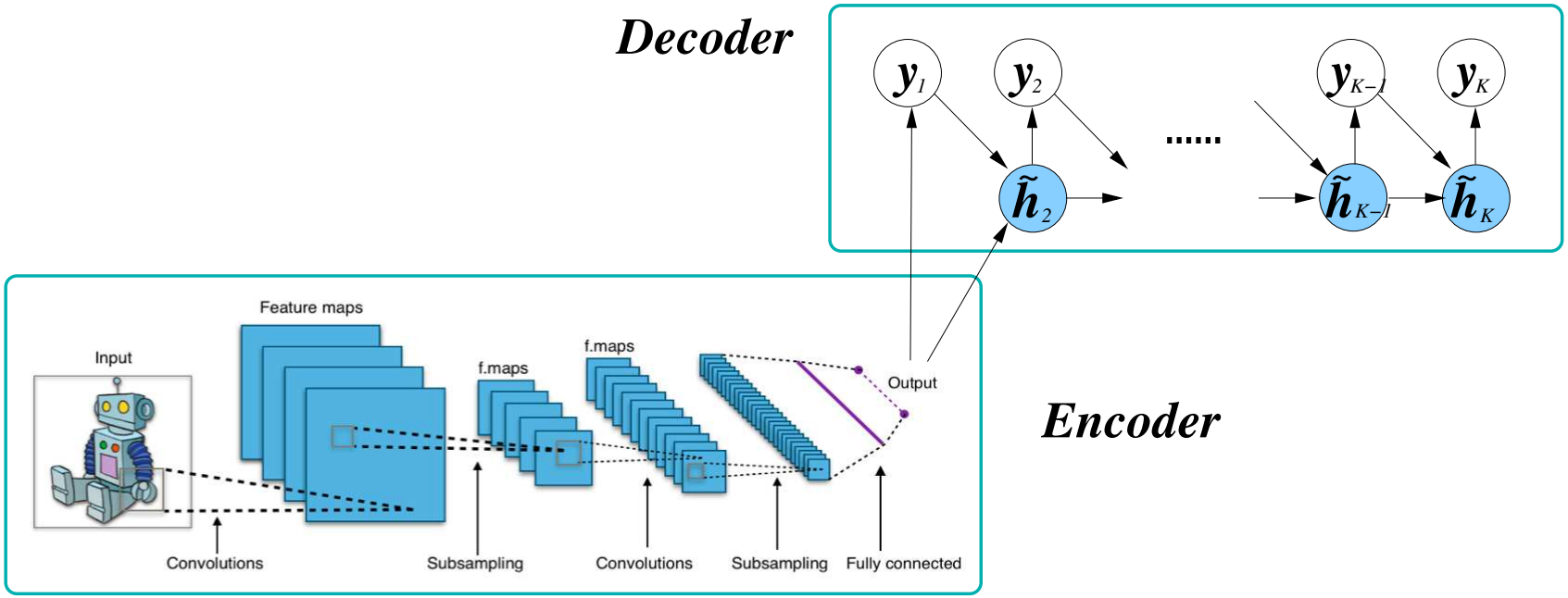
Single Input to Sequence

Image Captioning

- Given image, $\boldsymbol{x}$, generate caption $\boldsymbol{w}_{1:K}$

- Images highly complex
  - what's most relevant?
  - how to generate a caption?

Image Captioning



- Encode-image as vector use a deep convolutional network
  - generate caption using recurrent network (RNN/LSTM)
  - all parameters optimised (using example image captions)

End-to-End Systems

- Interesting to examine output of the encoder
  - does not have to be interpretable
  - just convey information from image to caption generation

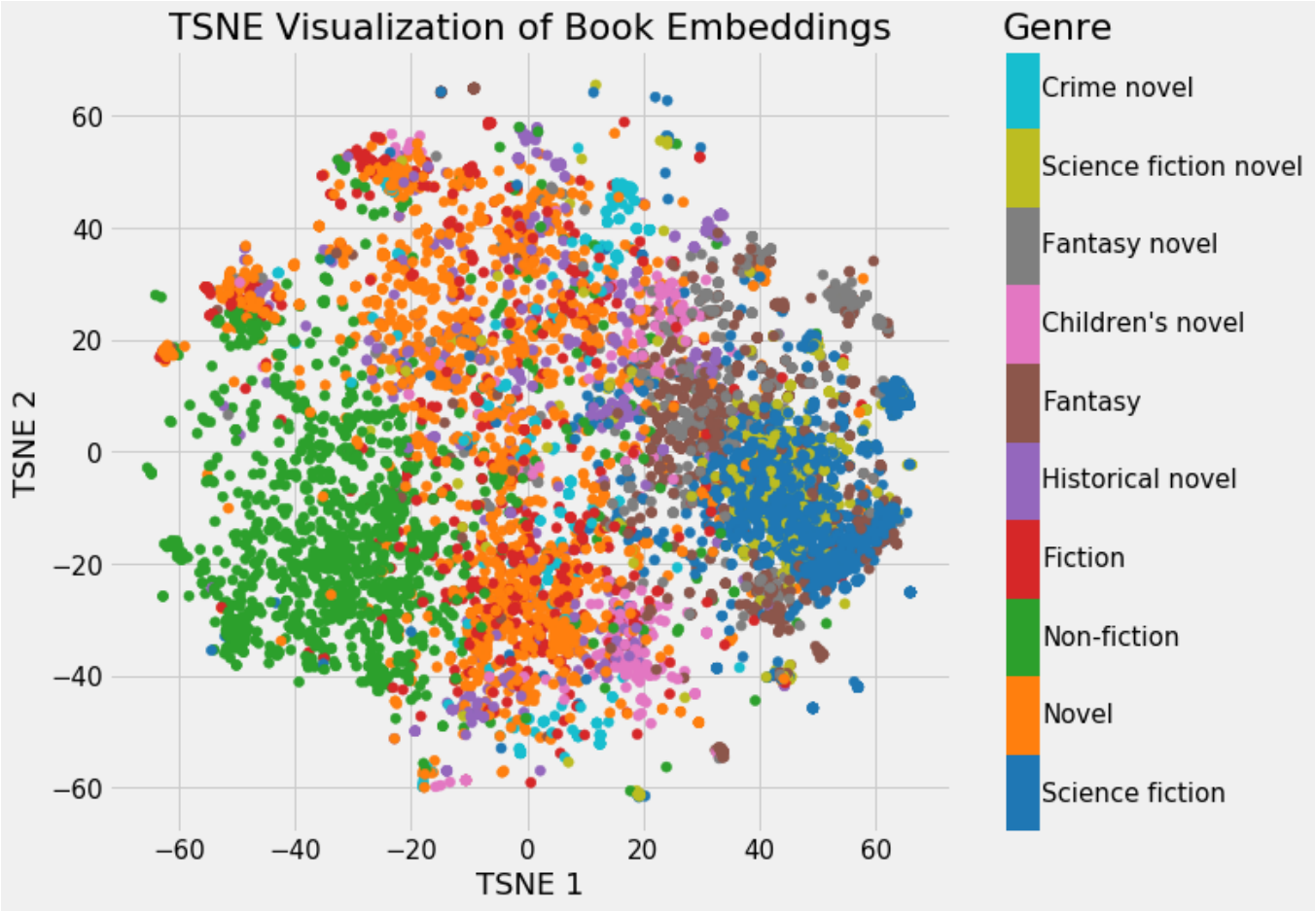
- An example of end-to-end training

  train the system to complete task!

  - minimises need for human expertise ...
  - no need to decide which aspect of image is important

Word Embeddings

Book Embeddings



Word2Vec [15]

- First stage of many text processing stages is word embedding

$$\underset{\text{word}}{w_i} \rightarrow \underset{\text{1-hot coding}}{\boldsymbol{t}_i} \rightarrow \underset{\text{(linear) embedding}}{\mathbf{W}\boldsymbol{t}_i = \boldsymbol{x}_i}$$

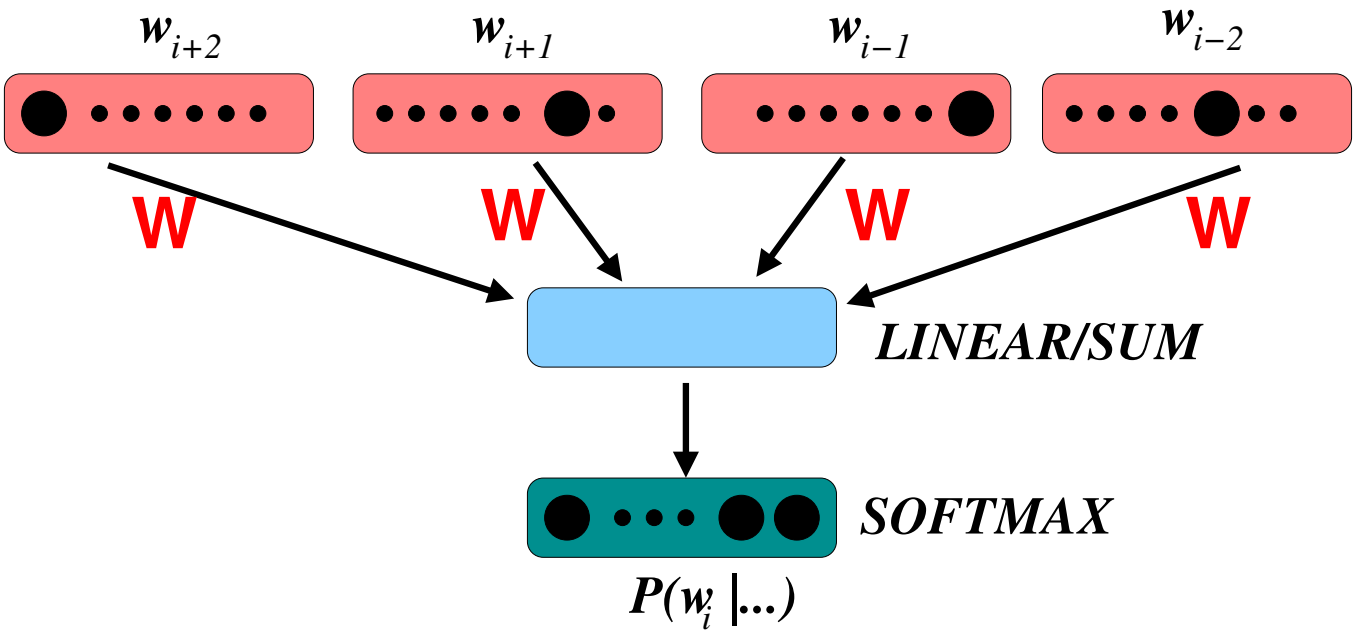
  - able to control dimensionality of $\boldsymbol{x}_i$
  - need to learn transformation matrix $\mathbf{W}$

- Could train on specific target training data
  - optimal transformation for target domain
  - often only limited data available

Train embedding just on general text data

- Able to fine-tune to specific domain - transfer learning

Word2Vec: Continuous Bag of Words (CBOW)



- Continuous Bag of words (CBOW): like feed-forward NN
  - learn shared word projections, **W**: maximise

$$P(w_i|w_{i+2}, w_{i+1}, w_{i-1}, w_{i-2})$$

  - other variants such as Skip N-Gram possible

Word2Vec Relationships

- System trained on Google 1-Billion word corpus
  - CBOW model used to train Word2Vec model
  - examine linear relationships between words

A	B	C	A-B+C
france	paris	rome	italy
king	man	woman	queen
did	do	give	gave

Context Dependent Word Embeddings [7]

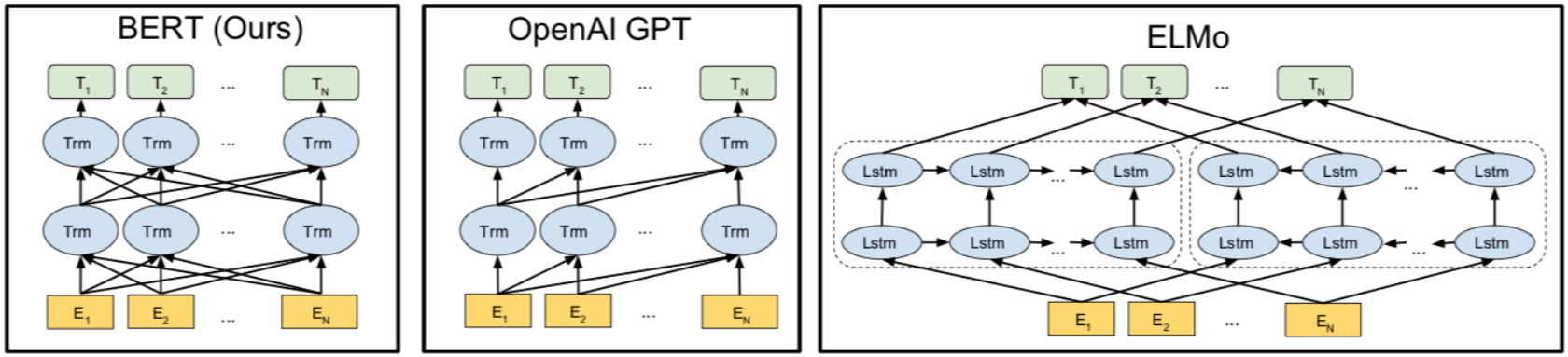
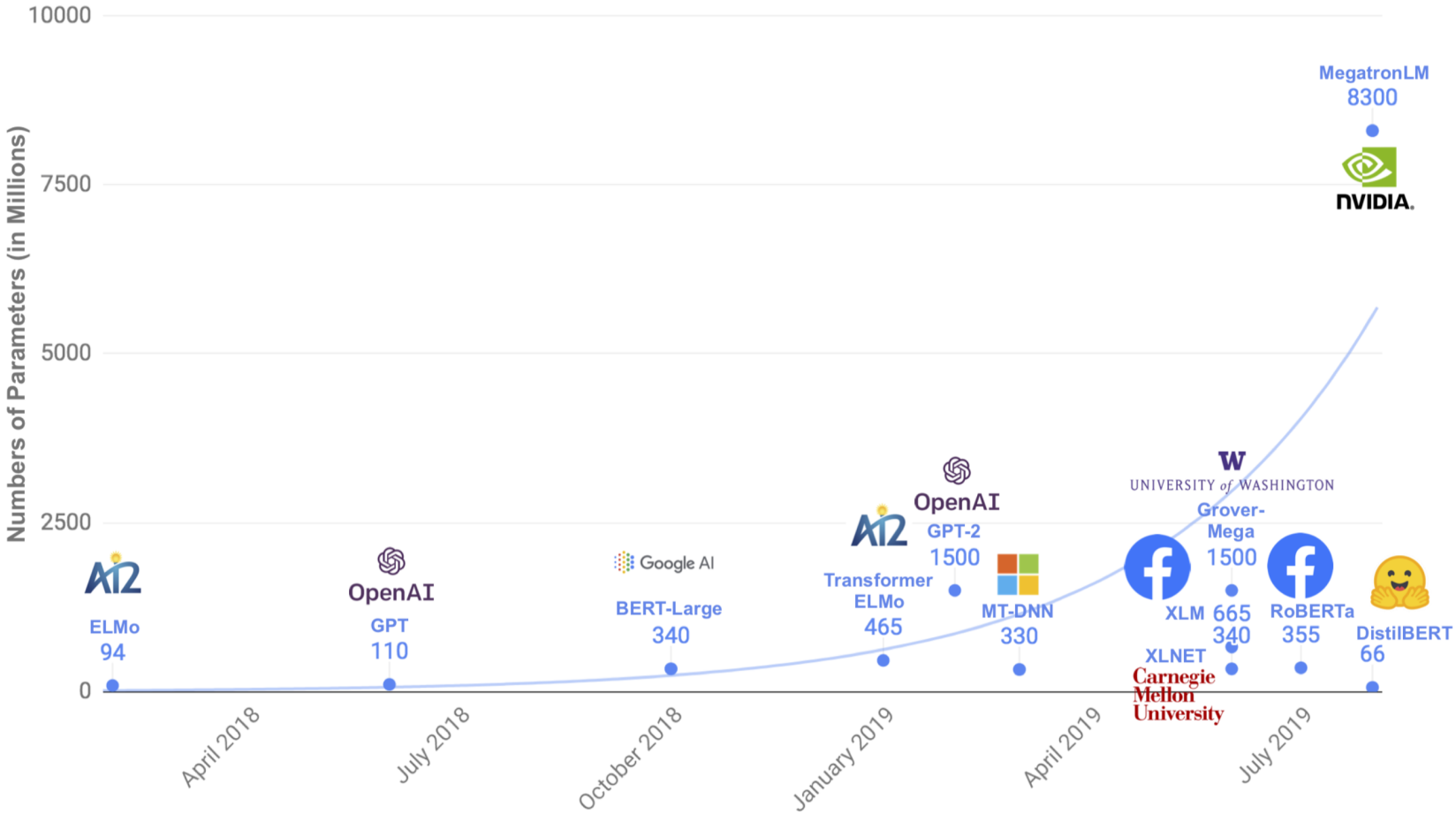


Figure: Figure from Google paper

- Interest in context dependent word embeddings
  - no unique embedding for a word
  - BERT: trained on large amounts of text data used

Recent Word Embeddings [18]



Language Modelling

Language Modelling

- Predict next word given observed sequence $\hat{w}_1, \ldots, \hat{w}_{i-1}$

$$\hat{w}_i \sim P(w_i|\hat{w}_1, \ldots, \hat{w}_{i-1}); \quad P(w_{i+1}|\hat{w}_1, \ldots, \hat{w}_i)$$

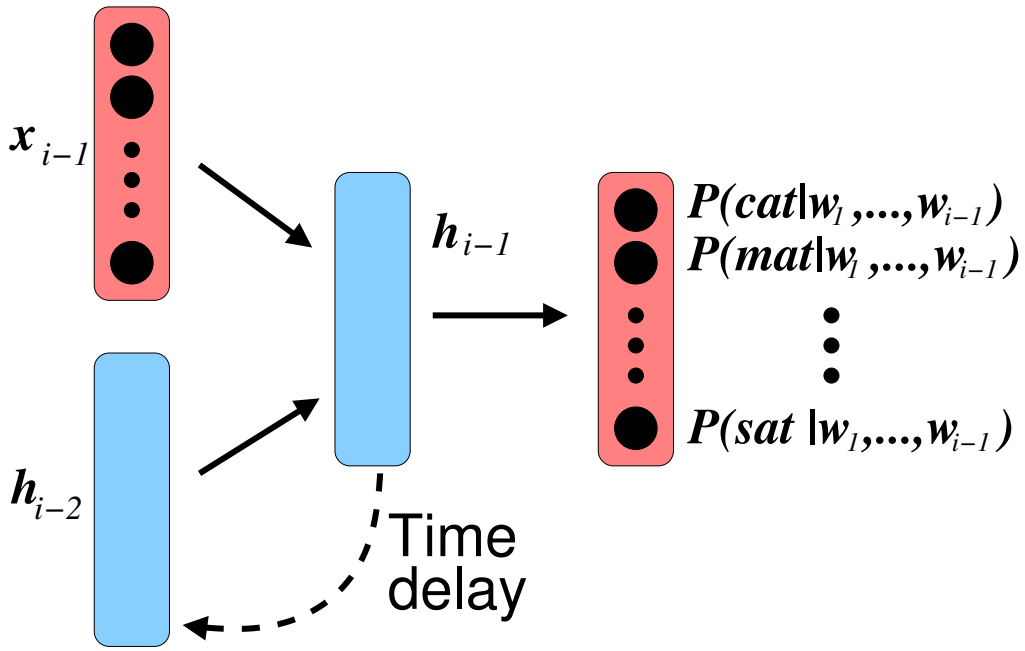
  - can repeat the process to generate a word sequence
  - simple auto-regressive data generation process
- Simple data generation example
  - initialise process with w_0 = <s>, sentence start symbol

<s> ???	$P(w_1|\texttt{<s>})$
<s> The ???	$P(w_2|\texttt{<s>}, \texttt{The})$
<s> The cat ???	$P(w_3|\texttt{<s>}, \texttt{The}, \texttt{cat})$
<s> The cat sat ???	$P(w_4|\texttt{<s>}, \texttt{The}, \texttt{cat}, \texttt{sat})$
<s> The cat sat on ???	$P(w_5|\texttt{<s>}, \texttt{The}, \texttt{cat}, \texttt{sat}, \texttt{on})$

- Can also be used to model word sequence distribution

$$P(\boldsymbol{w}_{1:L}) = P(w_1, \ldots, w_L) = \prod_{i=1}^{L} P(w_i|w_1, \ldots, w_{i-1})$$

RNN Language Models



- Use RNN history vector as word-history representation

$$P(\boldsymbol{w}_{1:L}) \approx \prod_{i=1}^{L} P(w_i|w_{i-1}, \boldsymbol{h}_{i-2}) = \prod_{i=1}^{L} P(w_i|\boldsymbol{x}_{i-1}, \boldsymbol{h}_{i-2}) \approx \prod_{i=1}^{L} P(w_i|\boldsymbol{h}_{i-1})$$

  - impact of word in history gradually decreases with distance from prediction

Guardian Article - 9th September 2020

Opinion Artificial intelligence (AI)

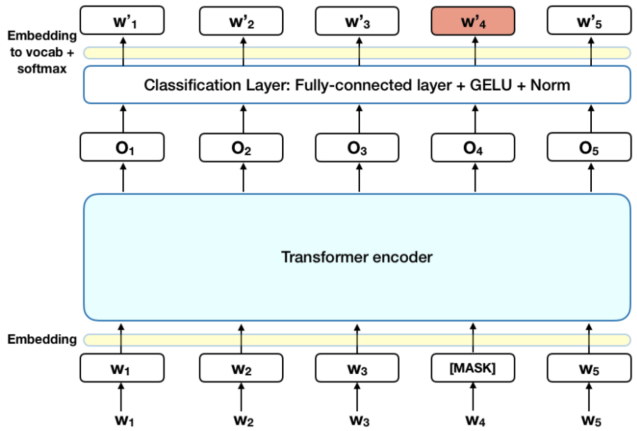
A robot wrote this entire article. Are you scared yet, human?

GPT-3

- Deep-learning has revolutionised language modelling
  - GPT-3 is a large language model from OpenAI
  - about 175 billion model parameters trained on 500 billion tokens
- Use sequence-to-sequence transformer architecture

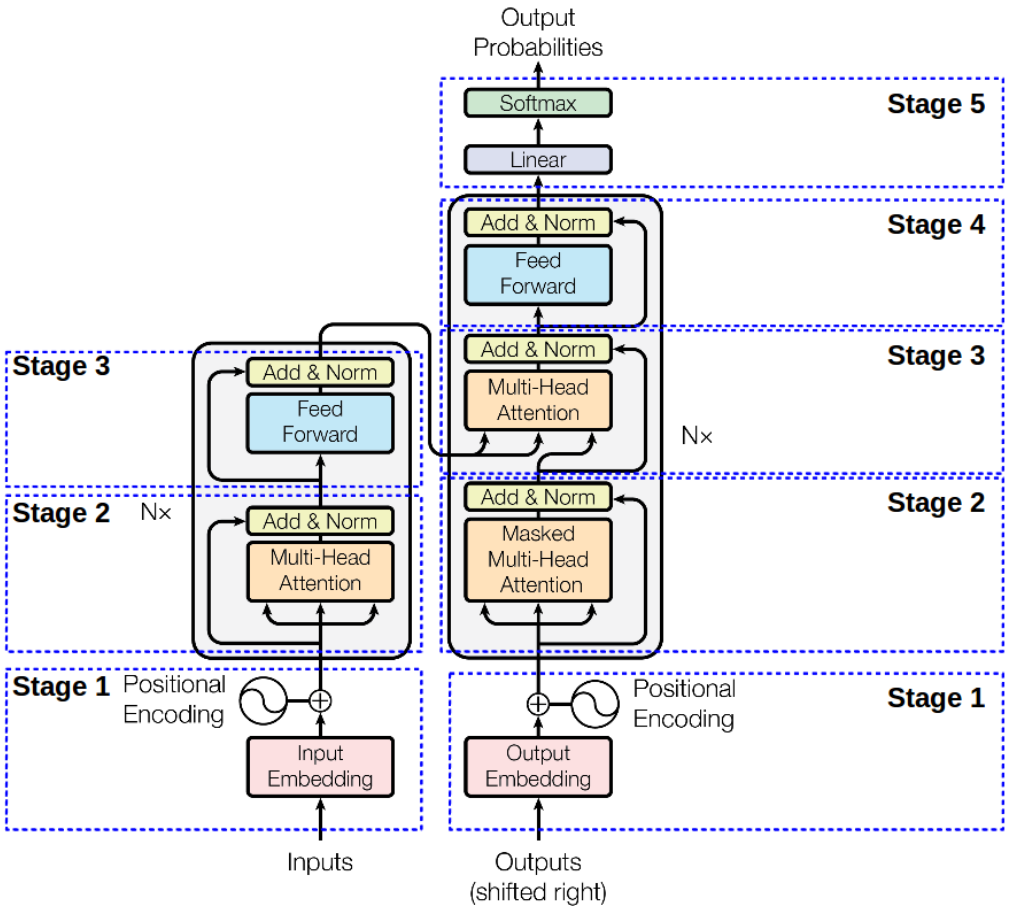
Masked Language Model: Start of BERT [7]

- Rather then predicting the next word in the sequence
  - predict one or more masked words in the complete sequence
  - uses infomation from complete sequence



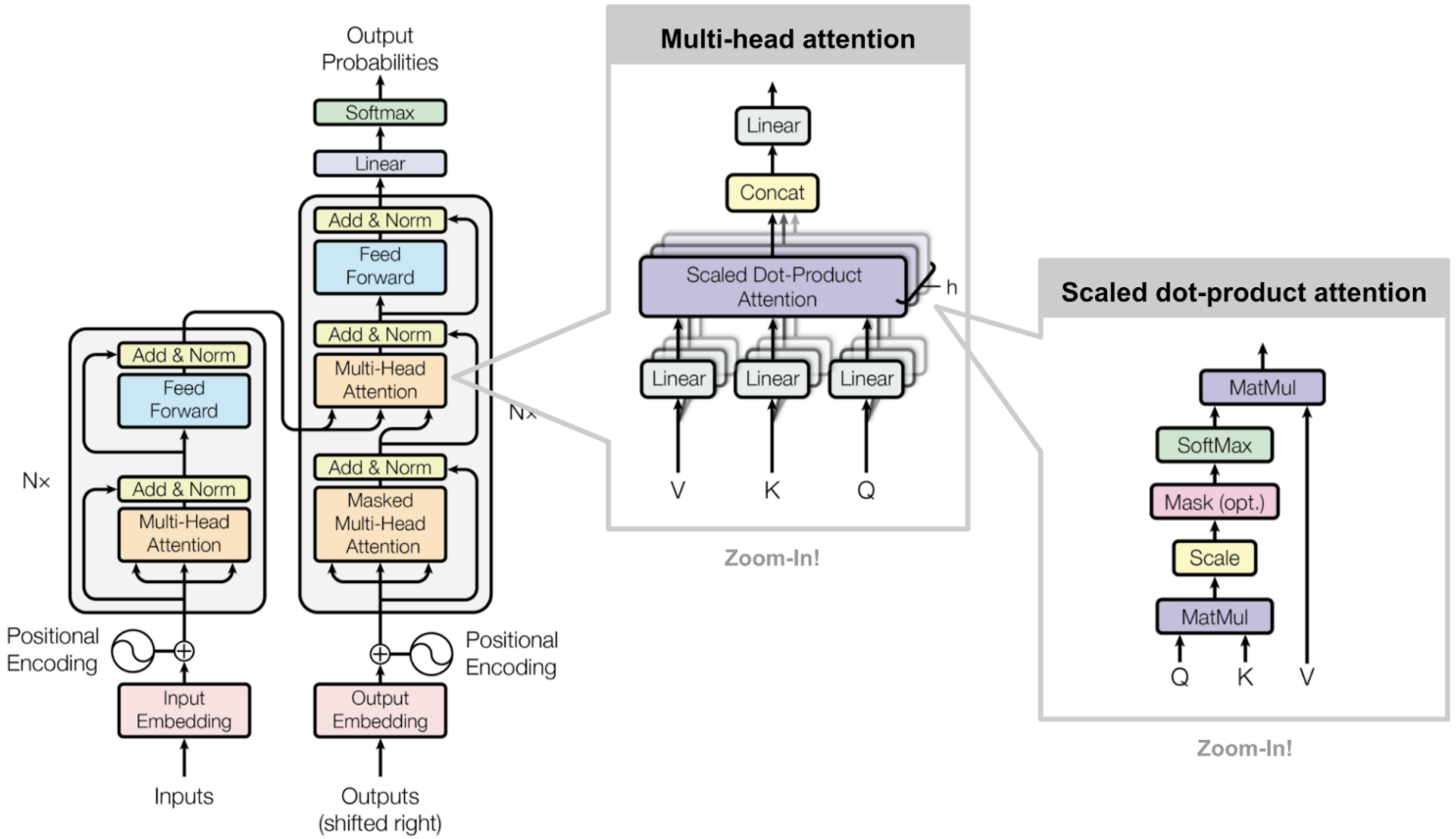
- Illustration above - fourth word is masked and predicted
  - (GELU - Gaussian Error Linear Unit, similar to a ReLU)
- Transformer encoder used as part of network
  - popular encoder, part of powerful seqeuence-to-sequence model
  - transformer decoder used in GPT-2/3

Transformer: Sequence-to-Sequence Model [21]



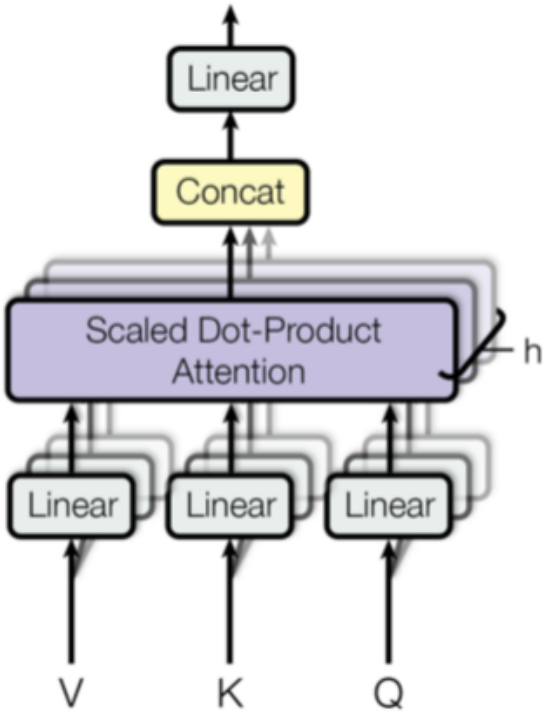
- Only based on self-attention: encoding/history/context
  - multiple transformer blocks for encoder and decoder

Transformer: Detail



- transform block comprises:
  - residual network plus layer norm (Add & Norm) acting on:
  - (1) Multi-Head Attention layer; (2) Feed-Forward network

Encoder Multi-Head Self-Attention



- Notation
  - Q: query for attention
  - K: key for attention - length d_k
  - V: input values $(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_L)$
- Scaled dot-product multi-head attention

$$\alpha_{\tau i}^{(j)} = \frac{1}{Z} \exp\left(\mathbf{k}_\tau^{(j)\mathsf{T}} \boldsymbol{q}_i^{(j)} / \sqrt{d_k^{(j)}}\right)$$

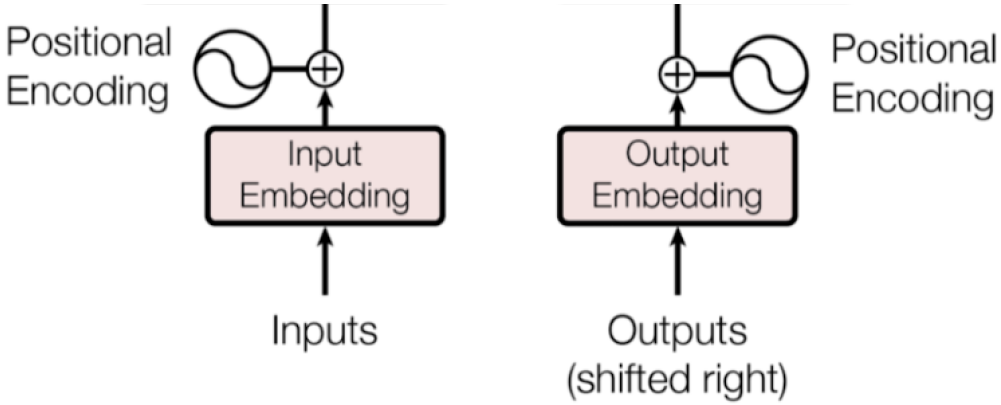
$$\boldsymbol{c}_i^{(j)} = \sum_{\tau=1}^{L} \alpha_{\tau i}^{(j)} \mathbf{W}_{\mathrm{v}}^{(j)} \boldsymbol{v}_\tau$$

$$\boldsymbol{c}_i = = \left[\begin{array}{ccc} \boldsymbol{c}_i^{(1)\mathsf{T}} & \ldots & \boldsymbol{c}_i^{(J)\mathsf{T}} \end{array} \right]^{\mathsf{T}}$$

- Self-attention $Q = K = V$

$$\boldsymbol{k}_i^{(j)} = \mathbf{W}_{\mathrm{k}}^{(j)} \boldsymbol{v}_i; \quad \boldsymbol{q}_i^{(j)} = \mathbf{W}_{\mathrm{q}}^{(j)} \boldsymbol{v}_i;$$

Position Encoding (reference)



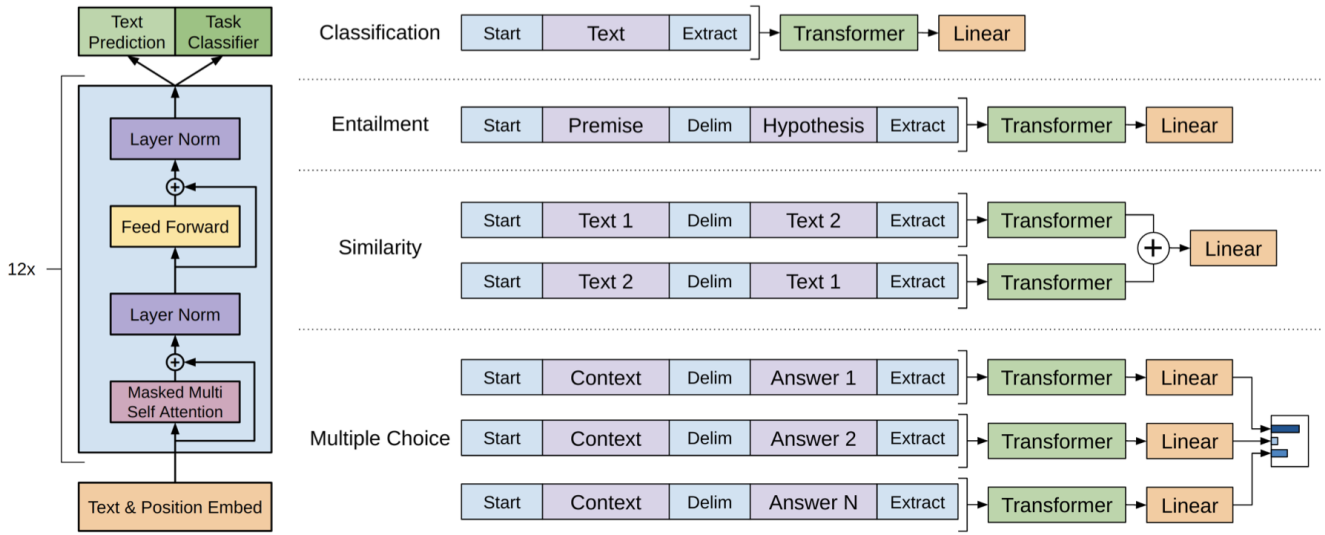
- Attention has no representation of position
  - but predictions should be sensitive to location
- Add position encoding vector word embedding
  - use sine and cosines of different frequencies

$$\text{pe}_{\tau,2i} = \sin\left(\tau/10000^{2i/d}\right)$$

$$\text{pe}_{\tau,2i+1} = \cos\left(\tau/10000^{2i/d}\right)$$

  - d is the number of nodes in a network layer
  - τ is the position of the word, i element of position vector

Transformer Decoder (reference) [13, 2]



- GPT-2/3 based on the transformer-decoder architecture
  - similar to decoder-side of the sequence-to-sequence transformer
  - autoregressive model with transformer units (no encoder)
  - train on multiple contiguous sentences (inc. sentence end tok)
- Pre-trained on large amounts of language model data
  - can be fine-tuned to any task of interest (RHS above)

[1] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 1171–1179. [Online]. Available: http://papers.nips.cc/paper/5956-scheduled-sampling-for-sequence-prediction-with-recurrent-neural-networks.pdf

[2] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020.

[3] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell," *CoRR*, vol. abs/1508.01211, 2015. [Online]. Available: http://arxiv.org/abs/1508.01211

[4] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," *CoRR*, vol. abs/1506.07503, 2015. [Online]. Available: http://arxiv.org/abs/1506.07503

[5] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[6] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," *CoRR*, vol. abs/1506.02216, 2015. [Online]. Available: http://arxiv.org/abs/1506.02216

[7] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds. Association for Computational Linguistics, 2019, pp. 4171–4186. [Online]. Available: https://doi.org/10.18653/v1/n19-1423

[8] A. Graves, "Sequence transduction with recurrent neural networks," *CoRR*, vol. abs/1211.3711, 2012. [Online]. Available: http://arxiv.org/abs/1211.3711

[9] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[10] K. M. Hermann, T. Kociský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," *CoRR*, vol. abs/1506.03340, 2015. [Online]. Available: http://arxiv.org/abs/1506.03340

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[12] A. M. Lamb, A. G. ALIAS PARTH GOYAL, Y. Zhang, S. Zhang, A. C. Courville, and Y. Bengio, "Professor forcing: A new algorithm for training recurrent networks," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 4601–4609. [Online]. Available: http://papers.nips.cc/paper/6099-professor-forcing-a-new-algorithm-for-training-recurrent-networks.pdf

[13] P. J. Liu, M. Saleh, E. Pot, B. Goodrich, R. Sepassi, L. Kaiser, and N. Shazeer, "Generating wikipedia by summarizing long sequences," 2018.

[14] L. Lu, X. Zhang, K. Cho, and S. Renals, "A study of the recurrent neural network encoder-decoder for large vocabulary speech recognition," in *Proc. INTERSPEECH*, 2015.

[15] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *CoRR*, vol. abs/1310.4546, 2013. [Online]. Available: http://arxiv.org/abs/1310.4546

[16] T. Robinson and F. Fallside, "A recurrent error propagation network speech recognition system," *Computer Speech & Language*, vol. 5, no. 3, pp. 259–274, 1991.

[17] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1," D. E. Rumelhart, J. L. McClelland, and C. PDP Research Group, Eds. Cambridge, MA, USA: MIT Press, 1986, ch. Learning Internal Representations by Error Propagation, pp. 318–362.

[18] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," 2019.

[19] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[20] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," *CoRR*, vol. abs/1505.00387, 2015. [Online]. Available: http://arxiv.org/abs/1505.00387

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. [Online]. Available: http://arxiv.org/abs/1706.03762