Master TRIED
Reconnaissance des formes et méthodes neuronales (US330X) - Neural Networks and Deep Learning

Nicolas Thome
Conservatoire Nationnal des Arts et Métiers (Cnam)
Laboratoire CEDRIC - équipe Vertigo

Outline

1. Text Representations & Embeddings
2. Recurrent Neural Networks (RNNs)
3. RNN Training
4. RNN Specific Architectures & Applications

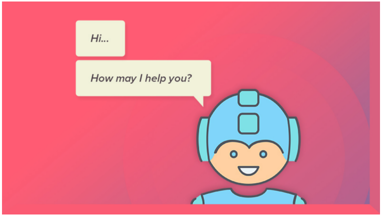
Natural Language Processing (NLP)

Short tour in NLP world

- At the boundary of:
  - Computer Science
  - Artificial Intelligence
  - Linguistic
- **Ultimate goal:** "understand" natural language for *e.g.* Question Answering

NLP tasks

- Text classification, *e.g.* sentiment analysis
- Machine Translation
- Image captioning (more next course)
- Chatbot, virtual assistant

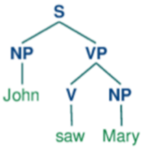


NLP sub-tasks

- Part-of-speech (POS) tagging
  - BiLSTM + CRF [Huang et al., 2015]: 97.6% accuracy *vs* ~ 96% HMM

The	grand	jury	commented	on	a	number	of	other	topics
DT	JJ	NN	VBD	IN	DT	NN	IN	JJ	NNS

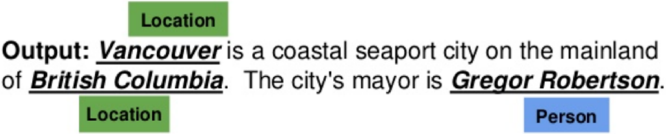
- Parsing
  - Seq2seq with attention [Vinyals et al., 2015]: ~ 93% accuracy



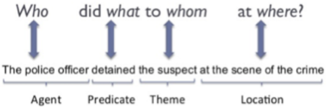
Symbol	Meaning	Example
S	sentence	*the man walked*
NP	noun phrase	*a dog*
VP	verb phrase	*saw a park*
PP	prepositional phrase	*with a telescope*
Det	determiner	*the*
N	noun	*dog*
V	verb	*walked*
P	preposition	*in*

NLP sub-tasks

- Named Entity Recognition (NER)



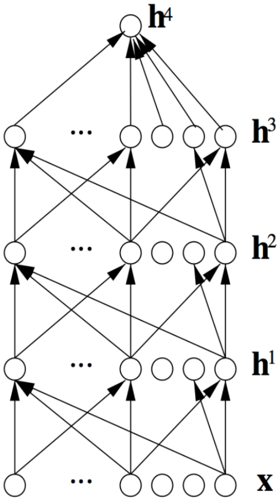
- Semantic Role Labeling (SRL): Who did what to whom?



Thematic Role	Definition
AGENT	The volitional causer of an event
EXPERIENCER	The experiencer of an event
FORCE	The non-volitional causer of the event
THEME	The participant most directly affected by an event
RESULT	The end product of an event
CONTENT	The proposition or content of a propositional event
INSTRUMENT	An instrument used in an event
BENEFICIARY	The beneficiary of an event
SOURCE	The origin of the object of a transfer event
GOAL	The destination of an object of a transfer event

NLP and Representation Learning

- Text: extracts "tokens", *i.e.* raw inputs
- How to represent raw inputs, *e.g.* characters, words, sentences?
- Handcrafted *vs* learned representations ⇒ **"deep embeddings"**



One-hot Representations

- Simplest encoding of text inputs: **one-hot representation**
- Binary vector of vocabulary size $|V|$, with 1 corresponding to term index
- $|V|$ small for chars (~ 10), large for words ($\sim 10^4$), huge for sentences
- Basis for constructing Bag of Word (BoW) Models

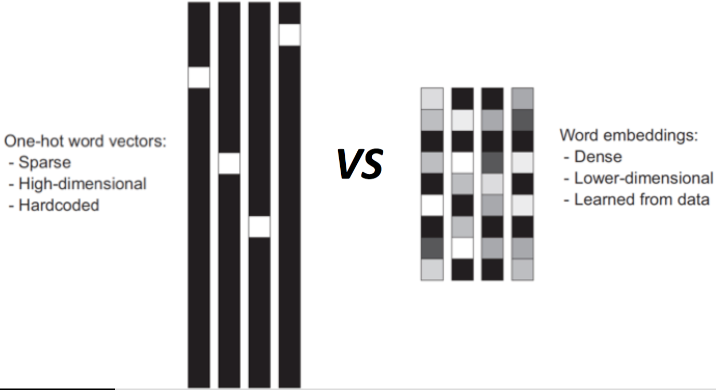
- Handcrafted feature used with ML shallow models, *e.g.* kernels methods
- Still very competitive for some NLP tasks, *e.g.* text topic classification
- Can be extended to (bags of) bi-grams for *e.g.* language identification

Beyond one-hot Representations

- Limitation: $\langle\ r(''motel'')\ ;\ r(''hotel'')\ \rangle = 0$

motel [0 0 0 0 0 0 0 0 0 0 1 0 0 0 0] AND
hotel [0 0 0 0 0 0 0 1 0 0 0 0 0 0 0] = 0

- Text embedding motivation: extract representation reflecting semantic similarities between text primitives ("Tokens")



Text Embeddings

- Learn mapping from one-hot encoding to a smaller vectorial space
- <u>**General idea:**</u> representing a word by means of its neighbors

“You shall know a word by the company it keeps”

(J. R. Firth 1957: 11)

- **Distributional Hypothesis:** One of the most successful ideas of modern statistical NLP

government debt problems turning into banking crises as has happened in

saying that Europe needs unified banking regulation to replace the hodgepodge

↖ These words will represent *banking* ↗

- Simplest historical strategy: SVD on one-hot encoded corpus

Text Embeddings

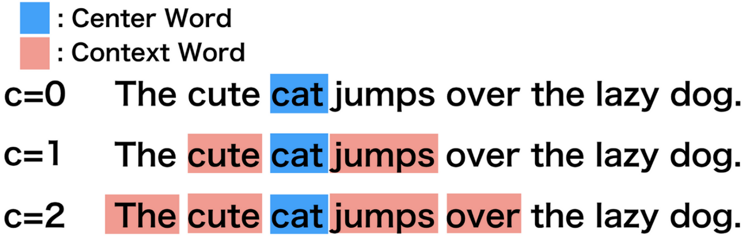
- Simplest historical strategy to represent context: co-occurrence matrices

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

- Dimensionality explosion $\mathcal{O}(|V|^2)$ $\Rightarrow$ memory & statistical robustness
- Use SVD to reduce dimension $\Rightarrow$ dense low-dimensional vector
    - Still, scalability with SVD

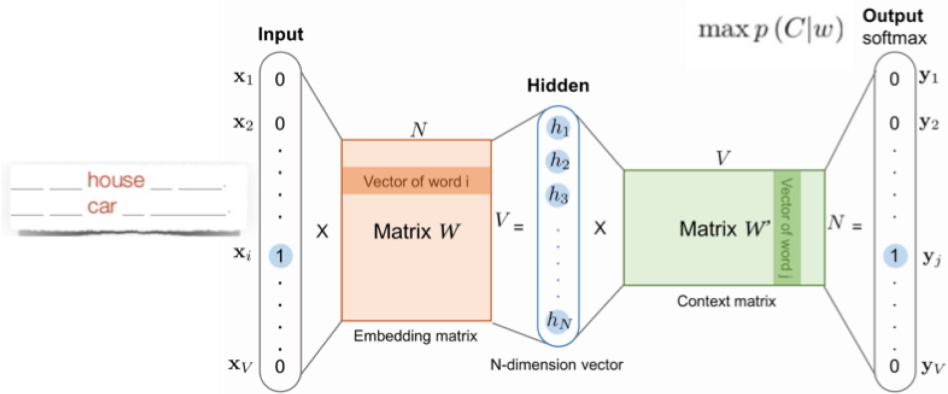
Text Embeddings: Word2vec [Mikolov et al., 2013]

- Modern approaches: directly learn low-dim text vectors
- **Word2vec** [Mikolov et al., 2013]: similar words $\Leftrightarrow$ similar contexts
  - Predict surrounding words (context) from central word: **CBoW**
  - Predict context from central word: **Skipgram**
- N.B.: cast unsupervised task as supervised one: "auto-supervision" (more next course) $\neq$ recontruction / ML, *e.g.* SVD



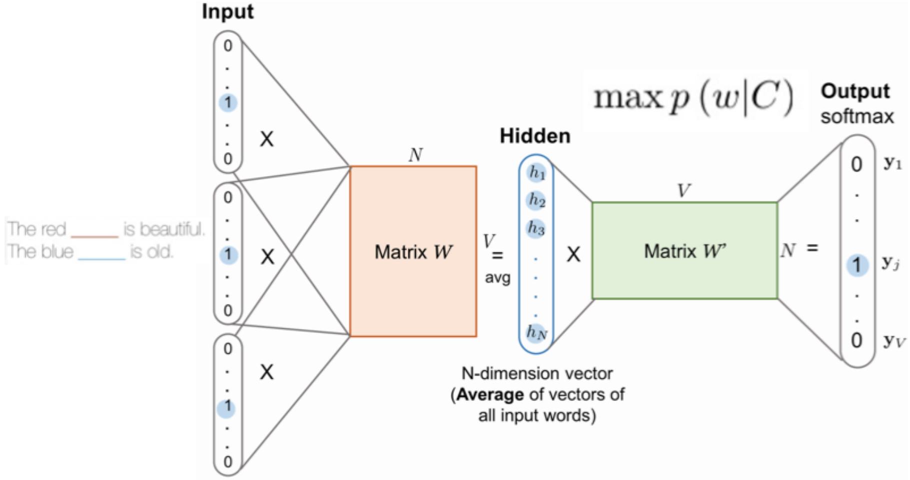
Word2vec [Mikolov et al., 2013]: Skipgram

- From a word w, one-hot encoded (size $|V|$) ⇒ infer its context C
- Project w into latent space h, size N ⇒ matrix W (select j^{st} column)
- Project h back into $|V|$ space ⇒ matrix W′ + soft-max
- Loss function: average cross-entropy for randomly context words



Word2vec [Mikolov et al., 2013]: CBoW

- From a context C one-hot encoded (size $|V|$) $\Rightarrow$ infer central word w
- Encode C into latent space h + average (or ~ project avg C $\Leftrightarrow$ BoW)
- Decode h back into $|V|$ space + soft-max



Word2vec [Mikolov et al., 2013]

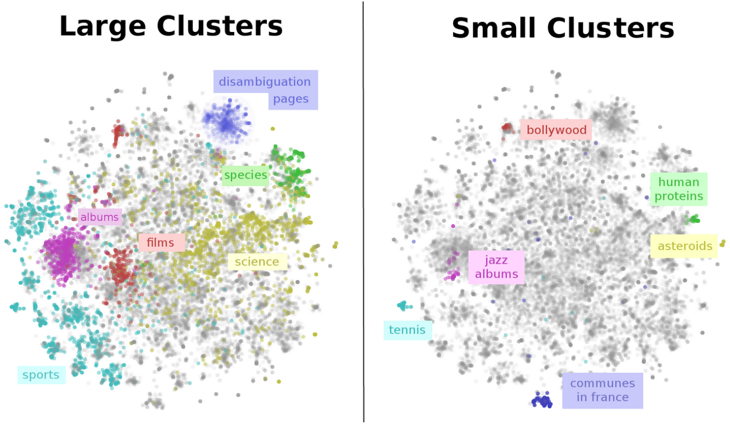
- Skipgram and CBoW trained with back-prop
- Soft-max : normalization over a huge vocabulary $|V|$:

$$\mathcal{L}(W) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} log\left(\frac{exp(v'_{w_{t+j}} v_{w_t})}{\sum_i exp(v'_{w_i} v_{w_t})}\right)$$ Options:

  - Hierarchical soft-max
  - Use sigmoid instead + negative sampling
- CBoW works well for frequent words, Skipgram for rare words
- Unsupervised: can be trained on huge generalist corpus
  - Transfer and fine-tuning possible on specific supervised tasks
    - Word2Vec and Glove ⇔ VGG or ResNet for vision
    - BUT only one layer transfer
- Extension of Skipgram for sentences Skip-Thought Vectors [Kiros et al., 2015]
  - predict the surroundings sentences of a given sentence
  - Extended to discriminative learning recently [Logeswaran and Lee, 2018] ⇒ faster training

Word2vec [Mikolov et al., 2013] Space Visualization

- Visualization of the latent space: t-SNE
- Semantic clustering of concept
- Ex with paragraph word2vec trained on wikipedia: see http://colah.github.io/posts/2015-01-Visualizing-Representations/

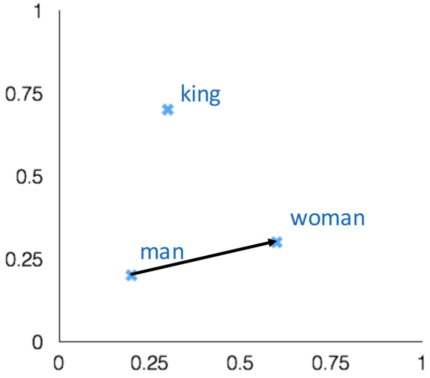


Analogies with Word Embeddings

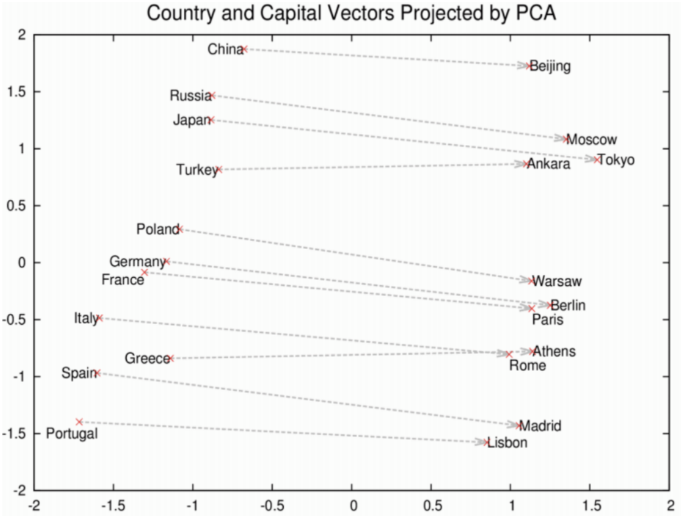
- Word Vector Analogies: $\boxed{\text{a:b :: c:?}}$, *e.g.* $\boxed{\text{man:woman:: king:?}}$
  ⇒ **map the relation between a and b to c**

- Assumption: can be done with simple algebraic operations (sum, subtraction): $r(c) + r(b) - r(a)$
- Disentangling in the learning representation space

$$d = \arg\max_i \frac{[r(c) + r(b) - r(a)]^T r(i)}{||r(c) + r(b) - r(a)||}$$



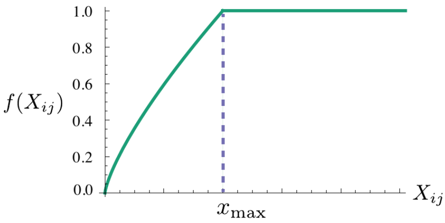
Analogies with Word Word2Vec: Example



Glove [Pennington et al., 2014]

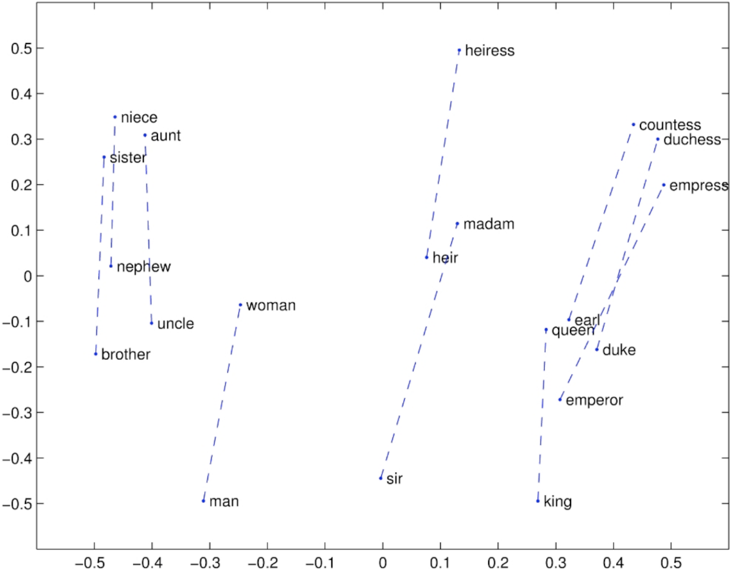
- GloVe: Global Vectors for Word Representation
- Based on co-occurrence matrix X $\Rightarrow$ leverage global context
  - Based on ratio of co-occurrence probabilities
- Explicitly enforcing semantic embedding, *i.e.* $w_i^T w_j \approx log(X_{ij})$:

$$J = \sum_{i,j=1}^{|V|} f(X_{ij}) \left(w_i^T w_j - log(X_{ij}) \right)$$

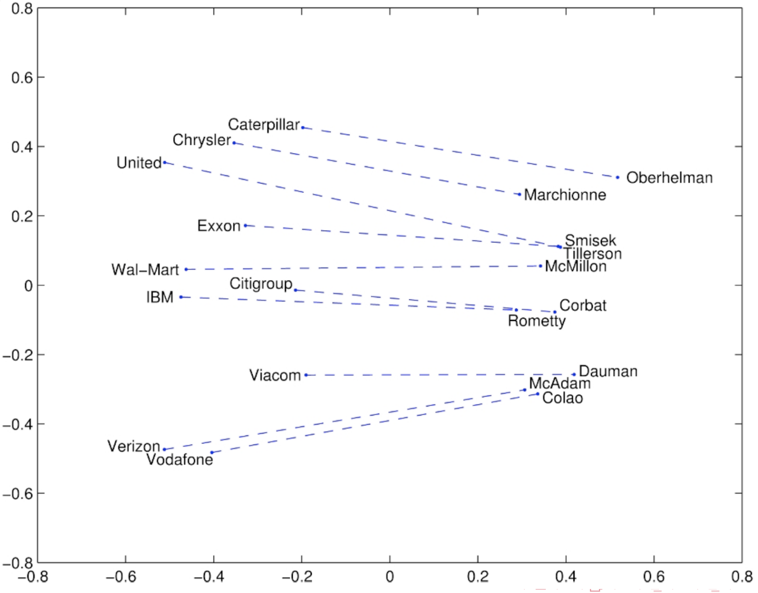


- w_i main word embedding, w_j context embedding (X computed on local windows)
- $f(X_{ij}) = \begin{cases} \left(\frac{x}{x_{max}}\right)^{\alpha} & \text{if } x \leq x_{max} \\ 1 & \text{otherwise} \end{cases}$

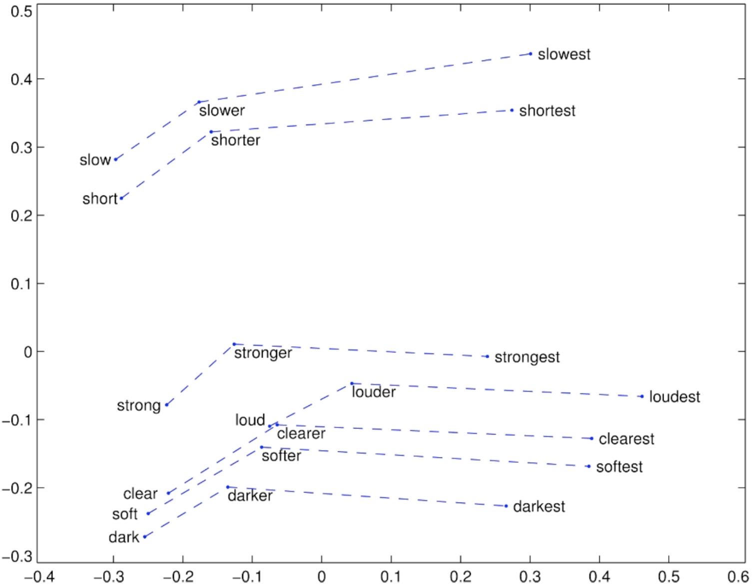
GlovE Analogies Examples: Man - Woman



GlovE Analogies Examples: Company - CEO



GlovE Analogies Examples: Superlatives

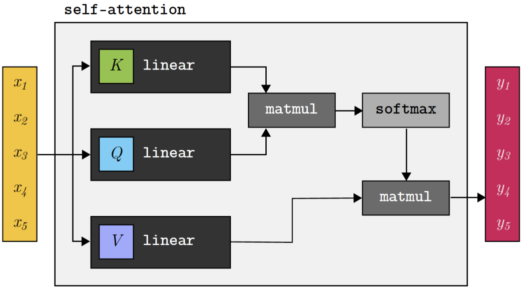


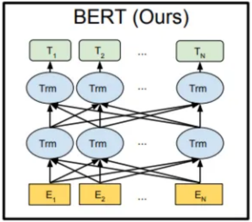
Other Recent Extensions

- FastText [Joulin et al., 2017]
- ELMo [Peters et al., 2018]
  - Standard word embeddings (word2Vec, Glove), word embedding context independent, *e.g.* "stick"
  - ELMo: an embedding based on the context it's used in - to both capture the word meaning in that context as well as other contextual information
- BERT [Devlin et al., 2018]: Bidirectional Encoder Representations from Transformers

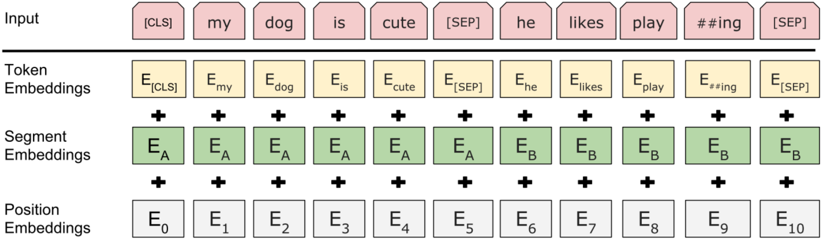
BERT: transformer model

- Only fully connected layers ($\neq$ ConvNets, RNNs)
- Self-attention (transformers), "attention is all you need" [Vaswani et al., 2017]
    - BERT: deep cascade of transformers *Tm*
    - Context-based embedding ($\neq$ Word2vec, Glove), bi-directional embedding ($\neq$ ELMO)





Transformer model: positional encoding



[CLS] : classification token

[SEP] : separate token

Pre-training corpus : BooksCorpus 、English Wikipedia

Token Embedding : WordPiece embeddings with a 30,000 token vocabulary.

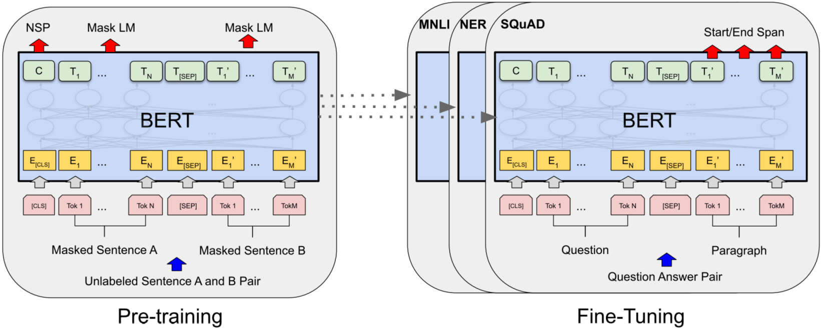
Segment Embedding : Learned embeddings belong to sentence A or sentence B.

Position Embedding : Learned positional embeddings.

Credit: Y. Fang

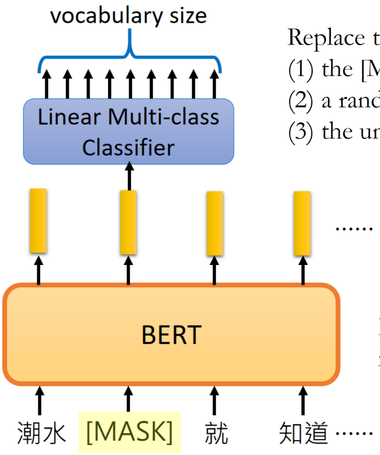
BERT: pre-training & fine-tuning

- Pre-training on unlabeled data: huge data (~ ImageNet in vision)
  - Next Sequence Prediction (NSP)
  - Masked Language Model (MLM)
- Fine-tuning on downstream task



Credit: Y. Fang

BERT: MLM



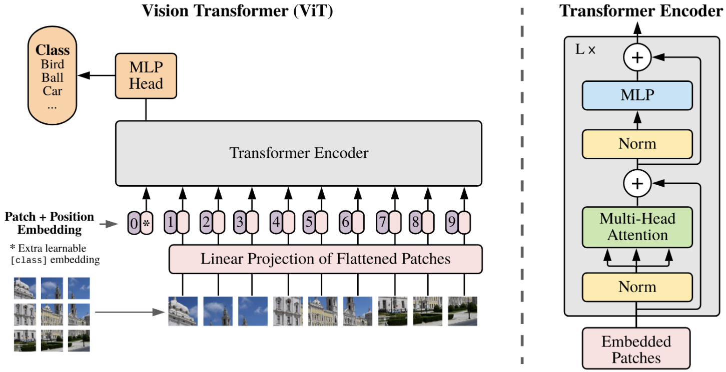
Replace the token with
(1) the [MASK] token 80% of the time.
(2) a random token 10% of the time.
(3) the unchanged i-th token 10% of the time.

Mask 15% of all WordPiece tokens
in each sequence at random for prediction.

Credit: Y. Fang

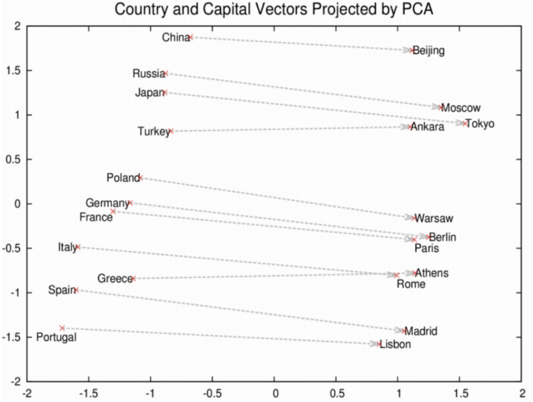
Transformers beyond NLP

- Since 2021: many attempt for using transformers beyond NLP
  - Main feature: global connections ≠ Convnets
  - Several success in vision: classification (ViT), detection (DETR), segmentation (SWIN), etc



Text Embeddings: Conclusion

- Nice semantic properties of the learned space
- Embeddings used as text dense representation as input of other models, *e.g.* Recurrent Neural Networks (RNNs)
  - Trained on huge corpus: universal text representations



Outline

1 Text Representations & Embeddings

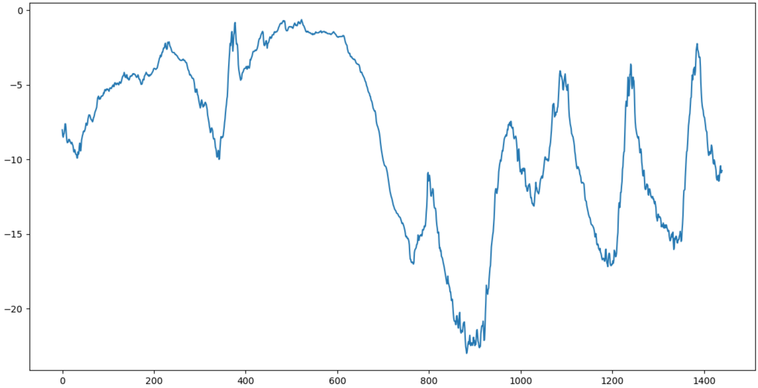
2 Recurrent Neural Networks (RNNs)

3 RNN Training

4 RNN Specific Architectures & Applications

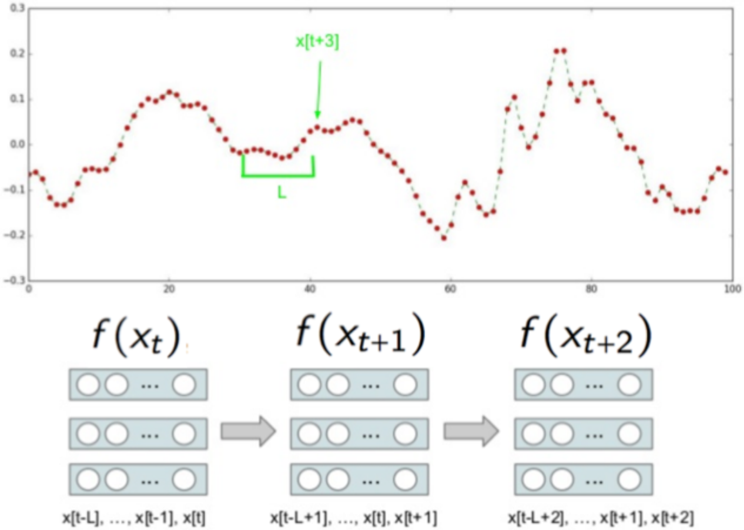
Motivation: Sequence processing

- Sequence: input $\{x_t\}_{t \in \{1;T\}}$
- Prediction $f(x_t)$ depends on $f(x_{t'})$ for $t' \leq t$
  - Text, time series, DNA, social media, *etc*

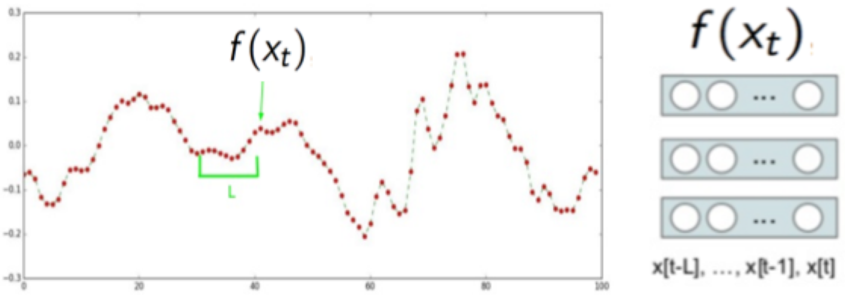


Sequence processing: options

- Fully connected network (FCN) on localized window, size L



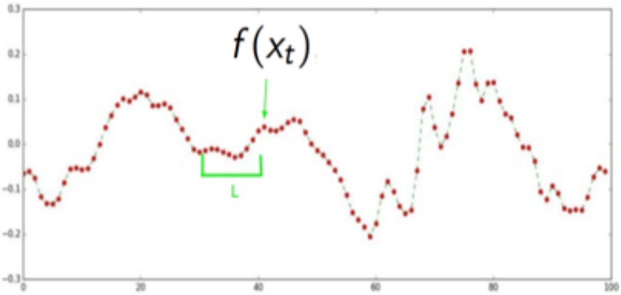
Sequence processing with FCNs: limitations

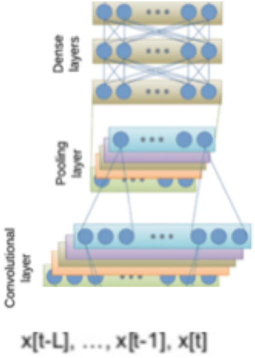


- ⊖ Increasing L ⇒ # parameter explosion!
- ⊖ Independent decisions between time steps
- ⊖ Cannot handle variable length L

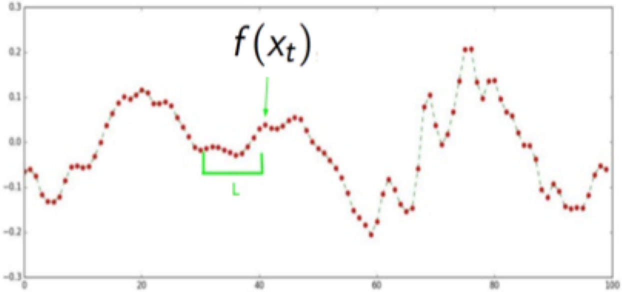
Sequence processing: options

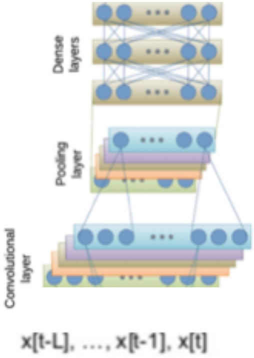
- Convolutional neural network (ConvNet) on localized window, size L





Sequence processing with ConvNets

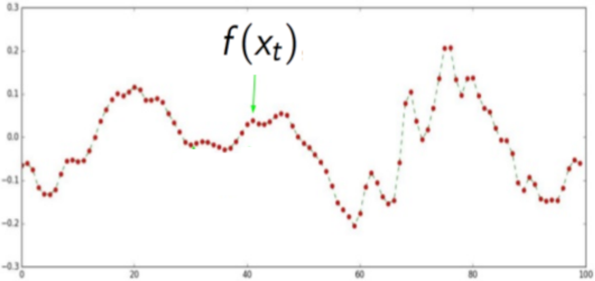


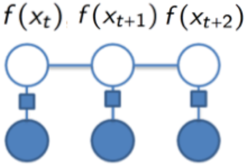


- ⊕ More compact than FCNs, locality, stability (see ConvNet course)
- ⊖ Cannot handle variable length L, or resolve to global pooling, maybe arbitrary

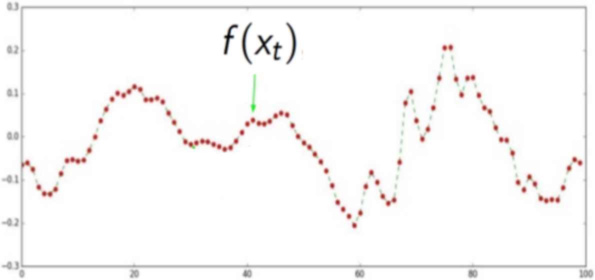
Sequence processing: options

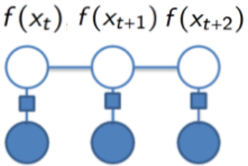
- Structured prediction: explicit modeling between $f(x_t)$ and $f(x_{t'})_{t' \leq t}$
- Markov models (Generative $P(x, y)$), Conditional Random Fields (CRF, discriminative) [Lafferty et al., 2001]





Sequence processing with CRFs

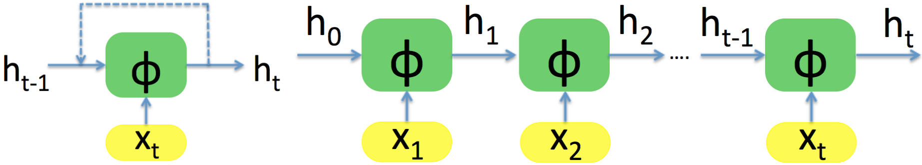




- ⊕ Can handle variable length L
- ⊖ Limited to linear predictors
- ⊖ Complex inference procedure, exact solutions need approximation
  - *e.g.* Markovian assumption: $f(x_T|x_t, t \leq T) = f(x_T|x_{T-1})$

Recurrent Neural Networks (RNNs) [Elman, 1990]

- Input sequence $\{x_t\}_{t \in \{1;T\}}$, $x_t \in \mathbb{R}^d$
- Internal RNN state $\{h_t\}_{t \in \{1;T\}}$, $h_t \in \mathbb{R}^l$
- **RNN Cell:** $h_t = \phi_t(x_t, h_{t-1})$
  - Loop, h_t depends on current x_t and previous state h_{t-1}
    - h_t: **memory of the network $\Leftrightarrow$ history up to time t**
  - In RNNs, function $\phi_t = \phi$ shared across time

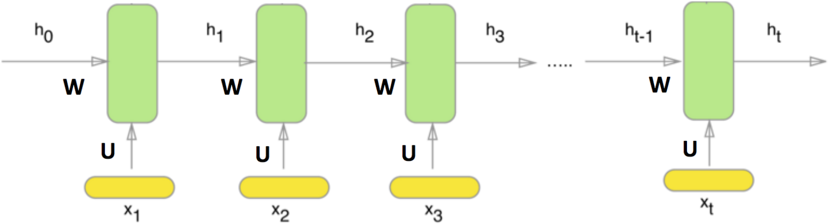


Recurrent RNN view

Unfolded RNN view

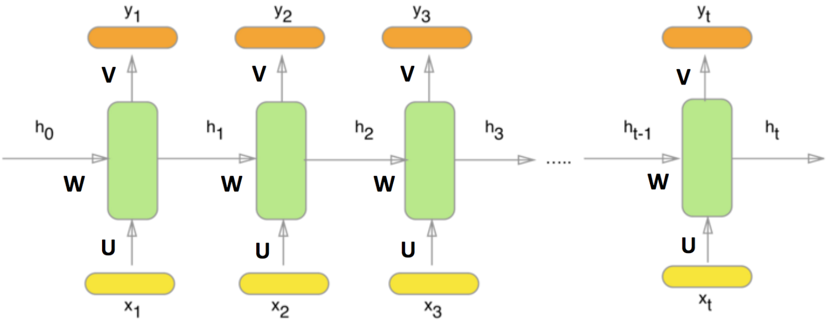
Recurrent Neural Networks (RNNs) [Elman, 1990]

- **RNN Cell:** $h_t = \phi(x_t, h_{t-1})$
  - ϕ: linear projection of x_t and h_{t-1}, *i.e.* fully connected layers
  - $h_t = f(Ux_t + Wh_{t-1} + b_h)$
    - U matrix size $l \times d$, W matrix size $l \times l$ (b vector size l)
    - $f \leftarrow tanh$ non-linearity



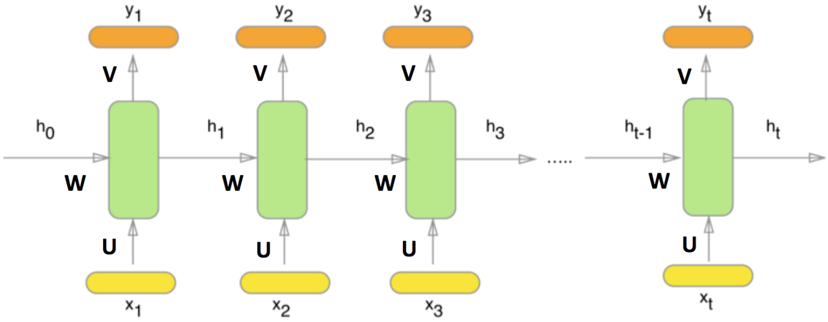
Recurrent Neural Networks (RNNs) [Elman, 1990]

- At each time step t, RNN output $\mathrm{y}_t = f'(\mathrm{V}\mathrm{h}_t + \mathrm{b}_y)$
  - $f' \leftarrow$ soft-max if $\mathrm{y}_t \leftrightarrow$ class probabilities

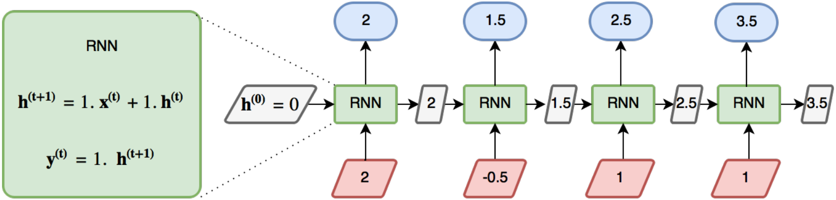


RNNs modeling power

- **Recap:** Feed-forward neural networks are universal function approximators [Cybenko, 1989]
- **Expressibility of the mapping between** $\{x_t\}_{t\in\{1;T\}}$ **and** $\{y_t\}_{t\in\{1;T\}}$**?**
  - RNNs are universal program approximators [Siegelmann and Sontag, 1995]
    - Can approximate any any computable function, *i.e.* Turing machine
  - RNNs can approximate any measurable sequence to sequence mapping [Hammer, 2000]

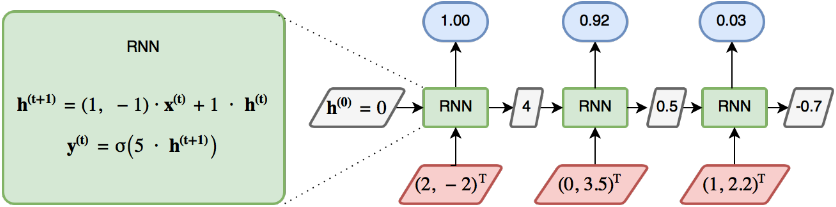


Example: Computing sum with RNNs



Example: Comparing dimension sum with RNNs

- Determining if the sum of the values of the first dimension is greater than the sum from the second dimension
  - $dim1 - dim2$ and then sum

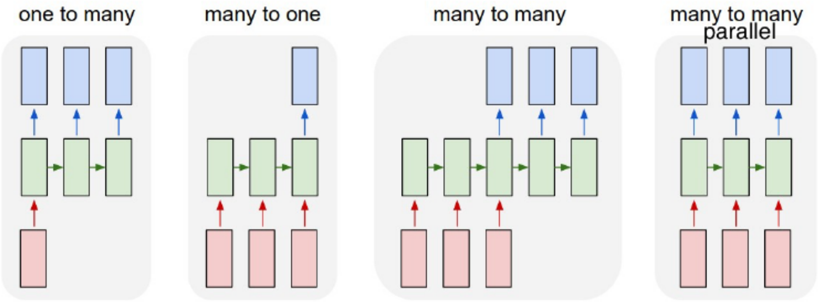


Outline

1. Text Representations & Embeddings
2. Recurrent Neural Networks (RNNs)
3. RNN Training
4. RNN Specific Architectures & Applications

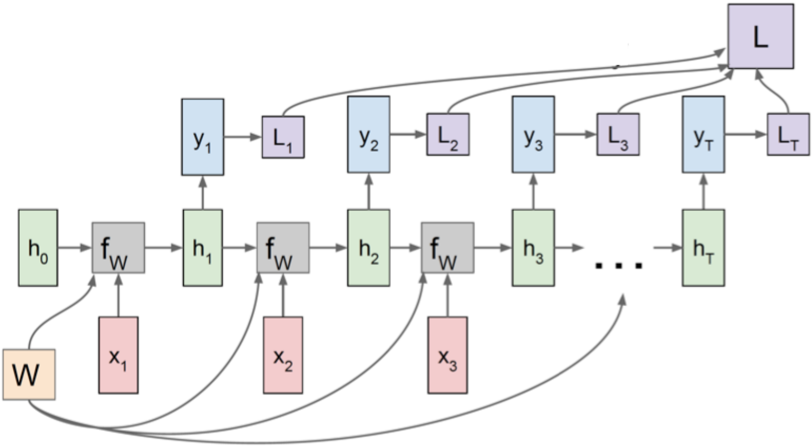
Training RNNs

- **RNN: mapping input sequence** $\{x_t\}_{t \in \{1;T\}}$ **into** $\{y_t\}_{t \in \{1;T\}}$
- **Different tasks** $\Leftrightarrow$ **different mappings**
  - **many-to-one:** sentiment classification, text generation (practical session), time series forecasting, VQA (next course)
  - **one-to-Many:** image captioning (next course)
  - **many-to-many parallel:** char-nn (predict next character)
  - **many-to-many (sub-part):** machine translation (text2text, spech2text), video classification (frame level)



Training RNNs: Formulation

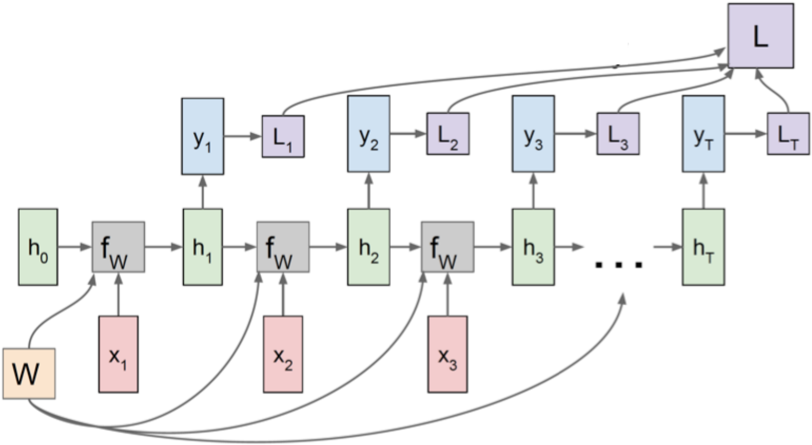
- Comparing output prediction $\{y_t\}_{t \in \{1;T\}}$ with supervision $\{y_t^*\}$
  - Task-dependent, *e.g.* only $\{y_T^*\}$ in many-to-one



Credit: Fei-Fei

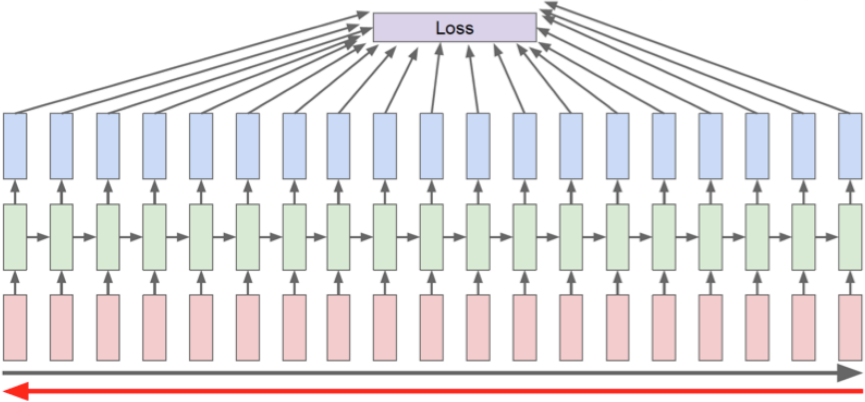
Training RNNs: Formulation

- Loss function at time t: $\mathcal{L}_t(y_t, y_t^*)$, *e.g.* cross-entropy (classification)
- Total loss function $\mathcal{L}(\{y_t\}, \{y_t^*\}) = \sum_{t=1}^{T} \mathcal{L}_t(y_t, y_t^*)$



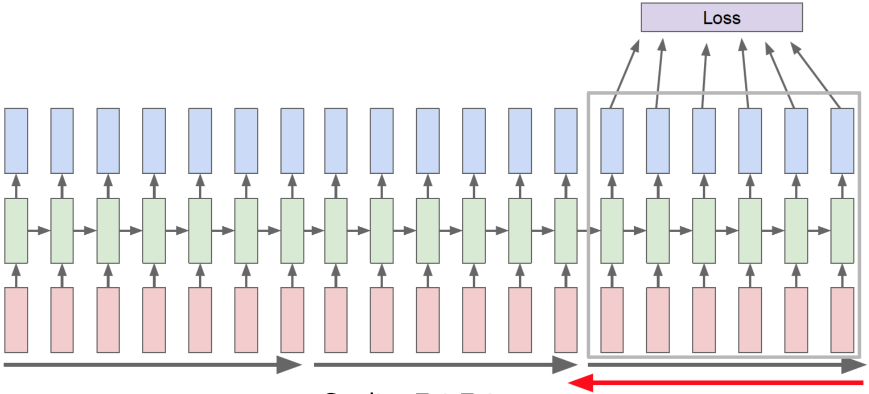
Credit: Fei-Fei

Back-Propagation Through Time (BPTT)



Credit: Fei-Fei

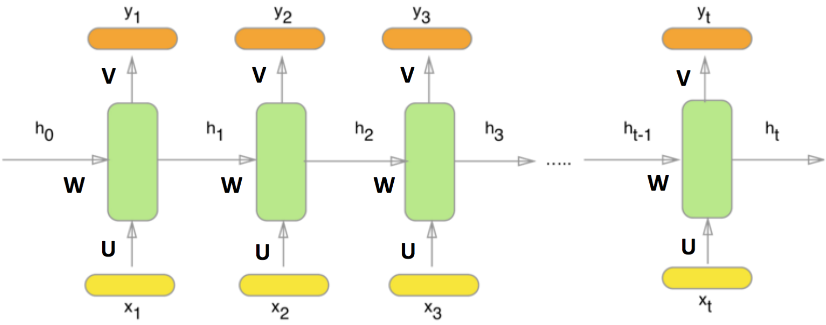
Truncated BPTT



Credit: Fei-Fei

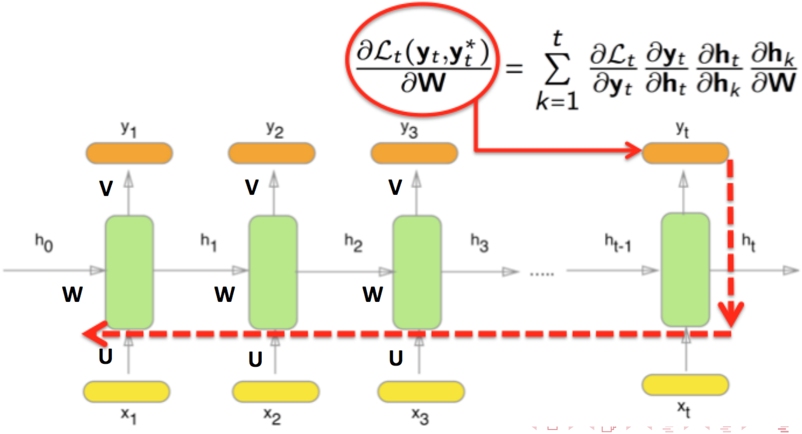
BPTT: Gradient Computation

- **BPTT:** computing gradient $\frac{\partial \mathcal{L}_t}{\partial W}$, $\frac{\partial \mathcal{L}_t}{\partial U}$, $\frac{\partial \mathcal{L}_t}{\partial V}$ (+biases)
- **Unfolded RNN:** same spirit as back-prop with fully connected networks (chain rule)
  - **BUT:** shared parameters W, U, V across time



BPTT: Gradient Computation

- **Shared parameters W, U, V across time**
  $\Rightarrow$ gradients depend on the whole past history
- Ex: for W: $\frac{\partial \mathcal{L}_t}{\partial W} = \sum_{k=1}^{t} \frac{\partial \mathcal{L}_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{\partial W}$



BPTT: Gradient Analysis

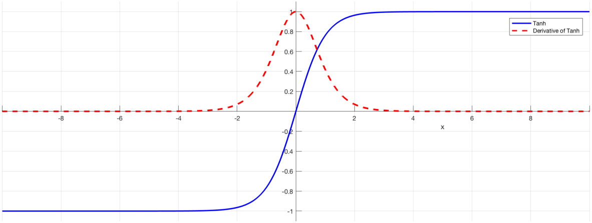
- $\frac{\partial \mathcal{L}_t}{\partial W} = \sum_{k=1}^{t} \frac{\partial \mathcal{L}_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \boxed{\frac{\partial h_t}{\partial h_k}} \frac{\partial h_k}{\partial W}$
- Chain rule (again): $\frac{\partial h_t}{\partial h_k} = \prod_{j=k+1}^{t} \frac{\partial h_j}{\partial h_{j-1}}$
- $h_t = f(Ux_t + Wh_{t-1} + b_h)$, *e.g.* f tanh
- Jacobian matrix $\frac{\partial h_j}{\partial h_{j-1}} = W^T diag[f'(h_{j-1})]$

$$\Rightarrow \text{Analyzing } \boxed{\left\|\frac{\partial h_t}{\partial h_k}\right\| = \left\|\prod_{j=k+1}^{t} W^T diag[f'(h_{j-1})]\right\|}$$

BPTT: Exploding and Vanishing Gradients

$$||\frac{\partial \mathrm{h}_t}{\partial \mathrm{h}_k}|| = ||\prod_{j=k+1}^{t} \mathrm{W}^T diag[f'(\mathrm{h}_{j-1})]|| \leq (\beta_w \beta_h)^{t-k}$$

- β_h for activation (tanh=1, sigmoid=0.25), β_w for W (largest eigenvalue)
  - $\beta_h \cdot \beta_w > 1 \Rightarrow$ exploding gradients
  - $\beta_h \cdot \beta_w < 1 \Rightarrow$ vanishing gradients
- True for any deep networks, exacerbated for RNNs



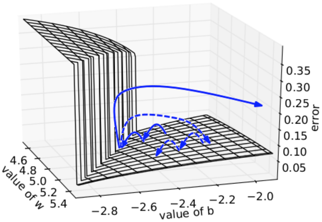
BPTT: Solutions for Exploding Gradients

- Use truncated BPTT, but smaller range dependencies
- Regularization, *e.g.* $||W||_2$ or $||W||_2$
- Simple common strategy: gradient clipping [Pascanu et al., 2013]

$\Rightarrow$ Exploding gradients relatively easy to detect and fix

Algorithm 1 Pseudo-code for norm clipping the gradients whenever they explode

$\hat{\mathbf{g}} \leftarrow \frac{\partial \mathcal{E}}{\partial \theta}$
if $\|\hat{\mathbf{g}}\| \geq threshold$ **then**
  $\hat{\mathbf{g}} \leftarrow \frac{threshold}{\|\hat{\mathbf{g}}\|} \hat{\mathbf{g}}$
end if



BPTT: Solutions for Vanishing Gradients

- Use Truncated BPTT, but smaller range dependencies
- Vanishing Gradient Regularization [Pascanu et al., 2013]
  - Controlling gradient magnitude evolution
- Using ReLU activation instead of tanh/sigmoid
- Using Hessian-free optimizer + damping [Martens and Sutskever, 2011]
  - approximate second order method
- Specific architectures/models, *e.g.* GRU/LSTM (see next)

BPTT: Bayesian Dropout [Gal and Ghahramani, 2016]

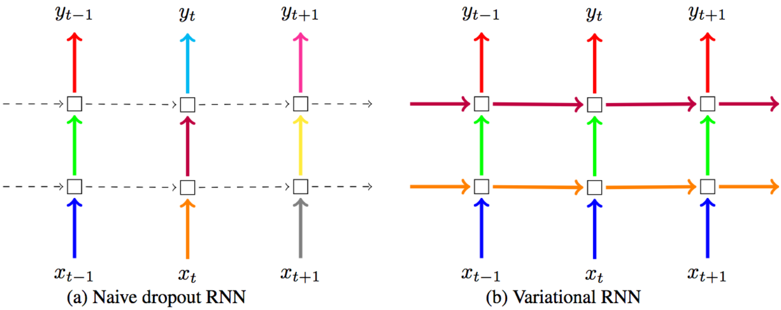


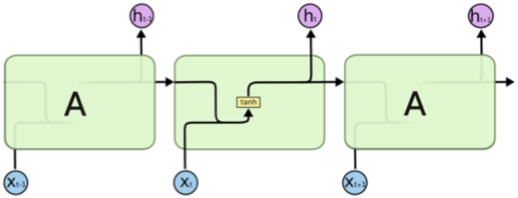
Figure 1: **Depiction of the dropout technique following our Bayesian interpretation (right) compared to the standard technique in the field (left).** Each square represents an RNN unit, with horizontal arrows representing time dependence (recurrent connections). Vertical arrows represent the input and output to each RNN unit. Coloured connections represent dropped-out inputs, with different colours corresponding to different dropout masks. Dashed lines correspond to standard connections with no dropout. Current techniques (naive dropout, left) use different masks at different time steps, with no dropout on the recurrent layers. The proposed technique (Variational RNN, right) uses the same dropout mask at each time step, including the recurrent layers.

Outline

1. Text Representations & Embeddings
2. Recurrent Neural Networks (RNNs)
3. RNN Training
4. RNN Specific Architectures & Applications

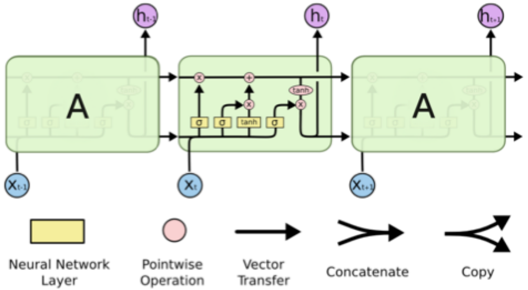
Robustness against vanishing gradients: LSTM

- Recap: Vanilla RNN cell



$$h_t = \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

- Long Short-Term Memory (LSTM) [Hochreiter and Schmidhuber, 1997]



$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

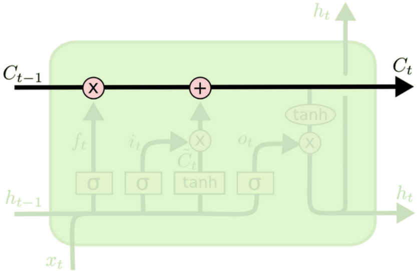
$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

Credit: C. Olah

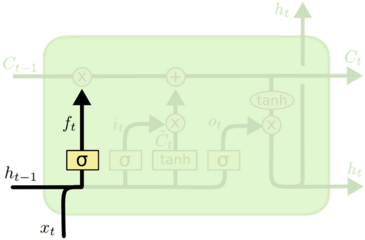
LSTM [Hochreiter and Schmidhuber, 1997]

- **Key modification:** Cell state C_t
- Easy for information (gradient) to flow along C_t path
- LSTM add/ remove information from the state C_t



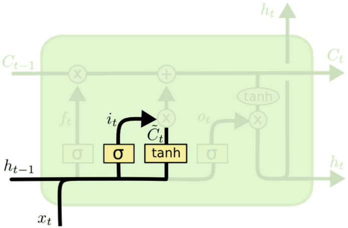
LSTM [Hochreiter and Schmidhuber, 1997]

- **Forget gate** f_t: whether to erase cell



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

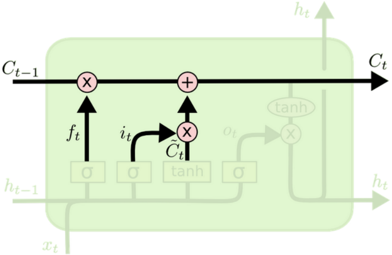
- **Input gate** i_t: whether to write to cell, **Gate** $\tilde{C}_t$: how much to write
  - $\sigma \in [0,1]$ (control gate, switch), *tanh* $\in [-1,1]$ (recurrent non-linearity)



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

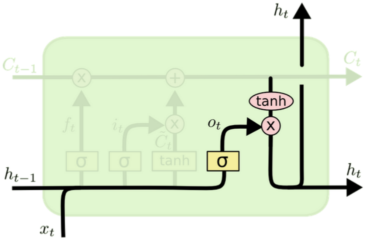
LSTM [Hochreiter and Schmidhuber, 1997]

- **Cell update**: key to LSTM



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

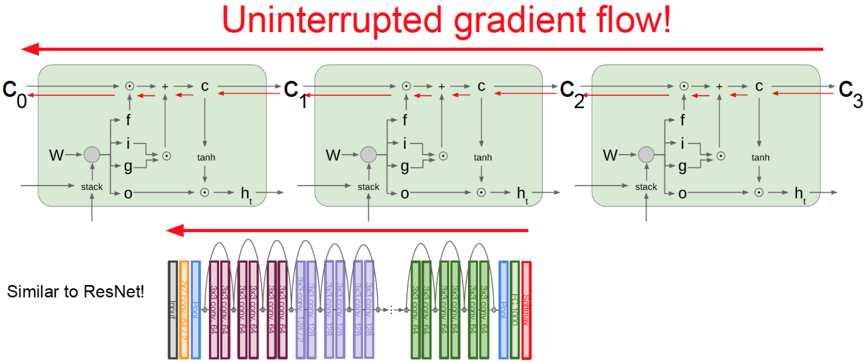
- **Output cell** o_t and internal state h_t (~ vanilla RNNs)



$$o_t = \sigma \left(W_o \; [h_{t-1}, x_t] \; + \; b_o \right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

LSTM [Hochreiter and Schmidhuber, 1997]: Gradient Flow

- Only elementwise multiplication and addition, no matrix multiply by W

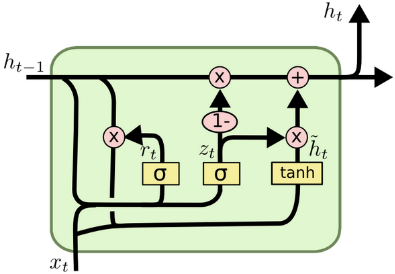


Credit: Fei-Fei

- In between: Highway Networks [Srivastava et al., 2015]
  $g = T(x, W_T)$
  $y = g \odot H(x, W_H) + (1 - g) \odot x$

Gated Recurrent Unit [Cho et al., 2014]

- LSTM popular variant: Gated Recurrent Unit (GRU) [Cho et al., 2014]
  - Combines forget and input gates into a single "update gate"
  - Merges the cell state and hidden state

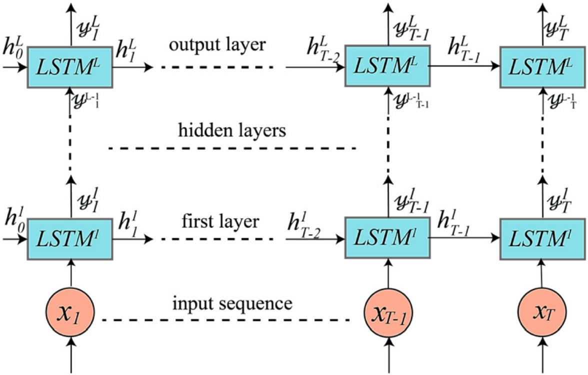


$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

- Simpler than LSTM, generally slightly inferior performances

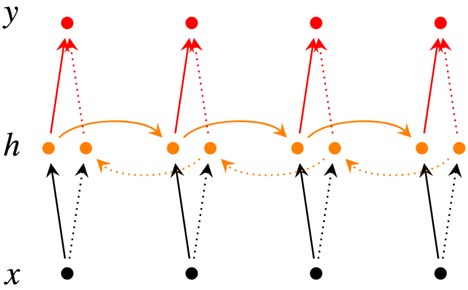
Deep RNNs

- Stacking RNN/ LSTM layers ⇒ learning more complex features
- Deep LSTM: very powerful, especially when stacked and made even deeper and if you have lots and lots of data



Bi-directionnal RNNs

- For classification, incorporate information from words both preceding and following



$$\overrightarrow{h}_t = f(\overrightarrow{W}x_t + \overrightarrow{V}\overrightarrow{h}_{t-1} + \overrightarrow{b})$$

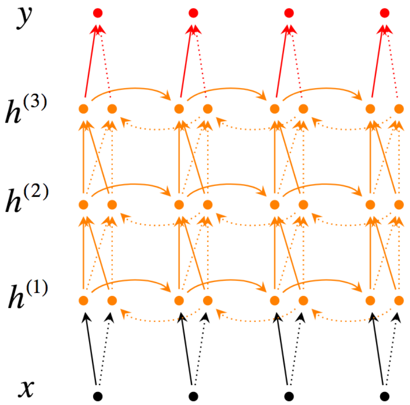
$$\overleftarrow{h}_t = f(\overleftarrow{W}x_t + \overleftarrow{V}\overleftarrow{h}_{t+1} + \overleftarrow{b})$$

$$y_t = g(U[\overrightarrow{h}_t;\overleftarrow{h}_t] + c)$$

$h = [\overrightarrow{h};\overleftarrow{h}]$ now represents (summarizes) the past and future around a single token.

Bi-directionnal RNNs

- Deep Bi-directionnal RNNs



$$\overrightarrow{h}_t^{(i)} = f(\overrightarrow{W}^{(i)} h_t^{(i-1)} + \overrightarrow{V}^{(i)} \overrightarrow{h}_{t-1}^{(i)} + \overrightarrow{b}^{(i)})$$

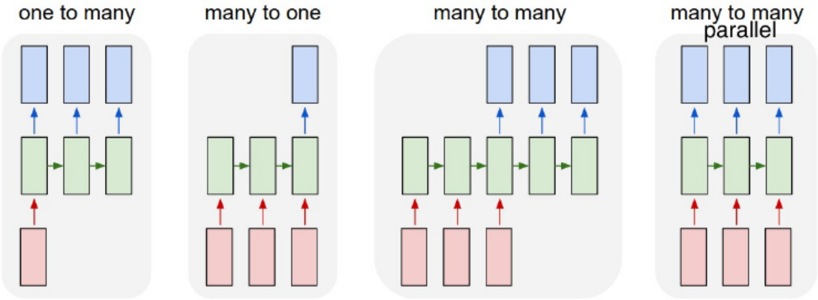
$$\overleftarrow{h}_t^{(i)} = f(\overleftarrow{W}^{(i)} h_t^{(i-1)} + \overleftarrow{V}^{(i)} \overleftarrow{h}_{t+1}^{(i)} + \overleftarrow{b}^{(i)})$$

$$y_t = g(U[\overrightarrow{h}_t^{(L)}; \overleftarrow{h}_t^{(L)}] + c)$$

Each memory layer passes an intermediate sequential representation to the next.

Applications

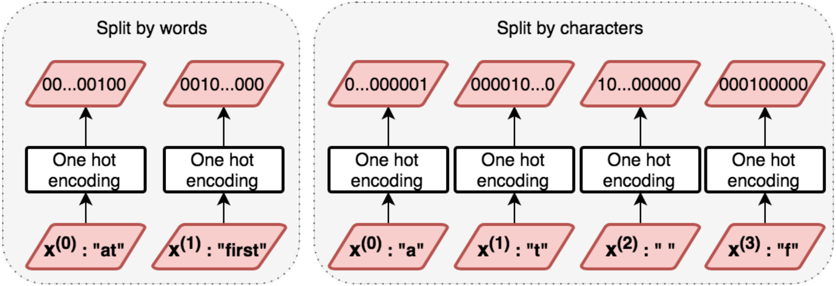
- RNN state of the art for many sequence processing applications



Applications: RNN for text processing

Deep NLP strategy:

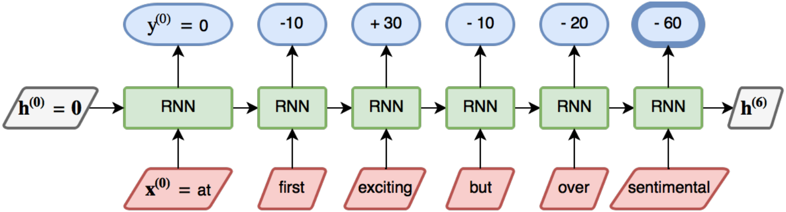
1. Extracts text input, "tokens", *e.g.* characters or words
2. One hot encoding of tokens



3. Split the text into a "temporal" sequence
4. **RNN to model the temporal structure**
   - Option: use an embedding layer on top of one-hot encoding

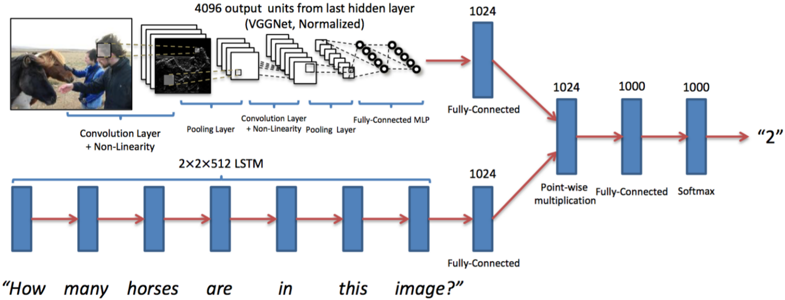
Applications: Many-to-one

- Sentiment classification
  - Input: "At first exciting but over sentimental"
    - Token ⇔ word
  - Output: -60 = Bad review



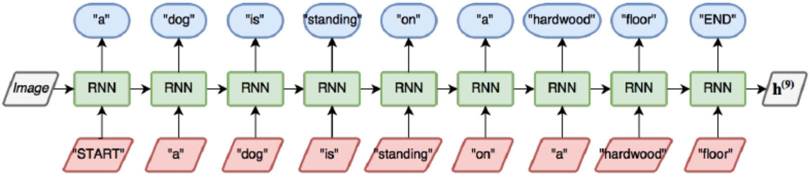
Applications: Many-to-one

- Visual Question Answering ⇒ next week
  - Token ⇔ word



Applications: One to Many

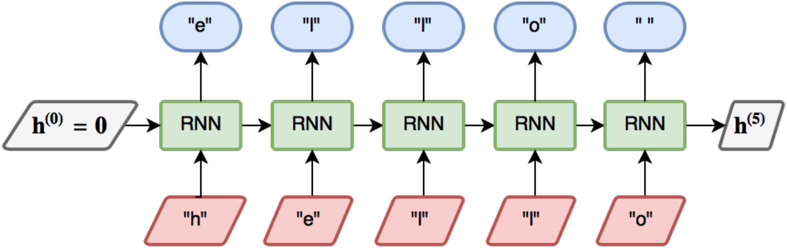
- Image captioning ⇒ next week
  - Token ⇔ word



[Karpathy and Li, 2015]

Applications: Sequence Generation

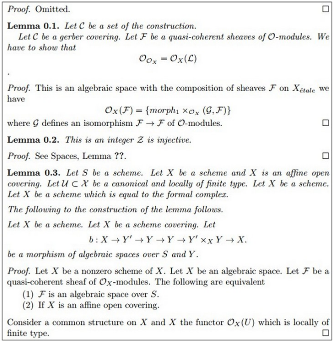
- Text (or music generation), *e.g.* Char-nn
  - Input sequence of characters (Token $\Leftrightarrow$ char)
  - Output: next character
- Many-to-many parallel
- In practice, trained with TBBTT $\Leftrightarrow$ many-to-one: predict next character from previous (K) chars



[Karpathy, 2015]

Text Generation - Char-nn

- Char-nn: applied to raw text, *e.g.* poetry (practical session)
  - Char-nn: learns to correctly spell a given language, although semantic meaning of sentences more challenging
  - Capacity to learn language structural/syntactical rules
    $\Rightarrow$ applications for generating source code, *e.g.* wikipedia pages, XML, Latex, linux source code (C), *etc*
    See here for other examples



Proof. Omitted. □

Lemma 0.1. *Let $\mathcal{C}$ be a set of the construction.*
Let $\mathcal{C}$ be a gerber covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $X_{étale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules. □

Lemma 0.2. *This is an integer $\mathcal{Z}$ is injective.*

Proof. See Spaces, Lemma ??. □

Lemma 0.3. *Let S be a scheme. Let X be a scheme and X is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let X be a scheme. Let X be a scheme which is equal to the formal complex.*

The following to the construction of the lemma follows.

Let X be a scheme. Let X be a scheme covering. Let

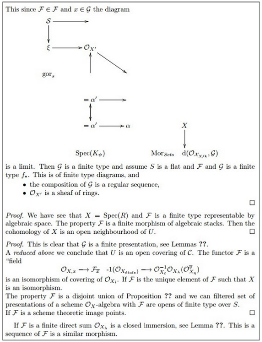
$$b : X \to Y' \to Y \to Y \to Y' \times_X Y \to X.$$

be a morphism of algebraic spaces over S and Y.

Proof. Let X be a nonzero scheme of X. Let X be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

(1) $\mathcal{F}$ is an algebraic space over S.
(2) If X is an affine open covering.

Consider a common structure on X and X the functor $\mathcal{O}_X(U)$ which is locally of finite type. □



This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram

is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

□

Proof. We have see that $X = \mathrm{Spec}(R)$ and $\mathcal{F}$ is a finite type representable by algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then the cohomology of X is an open neighbourhood of U. □

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas ??.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

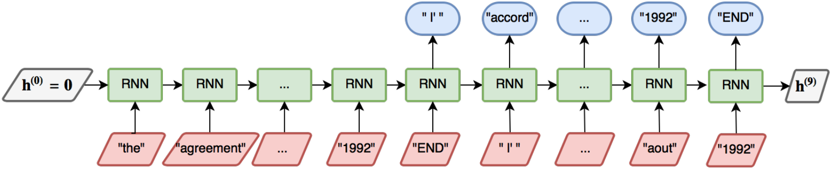
$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \ \text{-1}(\mathcal{O}_{X_{étale}}) \longrightarrow \mathcal{O}_{X_\ell}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{v}})$$

is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma ??. This is a sequence of $\mathcal{F}$ is a similar morphism.

Applications: Many to Many

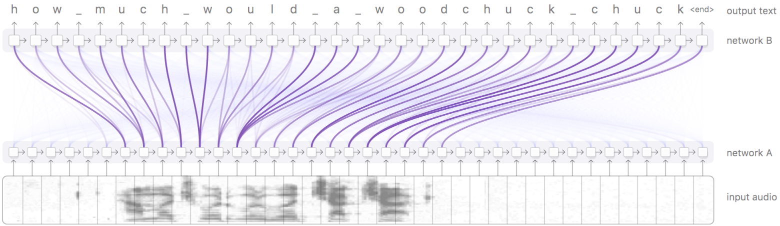
- Machine translation text2text
  - Input: "The agreement on the European Economic Area was signed in August 1992."
  - Output: "L'accord sur la zone économique européenne a été signé en août 1992."



[Bahdanau et al., 2014] [Olah and Carter, 2016]

Many to Many - Machine translation speech2text

- Machine translation speech2text
  - Input : Audio mp3 (speech utterance)
  - Output: "How much would a woodchuck chuck"



[Chan et al., 2015] [Olah and Carter, 2016]

Attention Mechanisms

- Used to focus the analysis of sequence on some specific inputs
  - Translation
  - Image captioning
  - VQA

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

References I

[Bahdanau et al., 2014] Bahdanau, D., Cho, K., and Bengio, Y. (2014).
Neural machine translation by jointly learning to align and translate.
CoRR, abs/1409.0473.

[Chan et al., 2015] Chan, W., Jaitly, N., Le, Q. V., and Vinyals, O. (2015).
Listen, attend and spell.
CoRR, abs/1508.01211.

[Cho et al., 2014] Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014).
Learning phrase representations using rnn encoder-decoder for statistical machine translation.
cite arxiv:1406.1078Comment: EMNLP 2014.

[Cybenko, 1989] Cybenko, G. (1989).
Approximation by superpositions of a sigmoidal function.
Mathematics of control, signals and systems, 2(4):303–314.

[Devlin et al., 2018] Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018).
BERT: pre-training of deep bidirectional transformers for language understanding.
CoRR, abs/1810.04805.

[Elman, 1990] Elman, J. L. (1990).
Finding structure in time.
COGNITIVE SCIENCE, 14(2):179–211.

[Gal and Ghahramani, 2016] Gal, Y. and Ghahramani, Z. (2016).
A theoretically grounded application of dropout in recurrent neural networks.
In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 1027–1035, USA. Curran Associates Inc.

References II

[Hammer, 2000] Hammer, B. (2000).
On the approximation capability of recurrent neural networks.
Neurocomputing, 31(1-4):107–123.

[Hochreiter and Schmidhuber, 1997] Hochreiter, S. and Schmidhuber, J. (1997).
Long short-term memory.
Neural Comput., 9(8):1735–1780.

[Huang et al., 2015] Huang, Z., Xu, W., and Yu, K. (2015).
Bidirectional LSTM-CRF models for sequence tagging.
CoRR, abs/1508.01991.

[Joulin et al., 2017] Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. (2017).
Bag of tricks for efficient text classification.
In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics.

[Karpathy, 2015] Karpathy, A. (2015).
The unreasonable effectiveness of recurrent neural networks.

[Karpathy and Li, 2015] Karpathy, A. and Li, F. (2015).
Deep visual-semantic alignments for generating image descriptions.
In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3128–3137.

[Kiros et al., 2015] Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., and Fidler, S. (2015).
Skip-thought vectors.
In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28*, pages 3294–3302. Curran Associates, Inc.

References III

[Lafferty et al., 2001] Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001).
Conditional random fields: Probabilistic models for segmenting and labeling sequence data.
In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Logeswaran and Lee, 2018] Logeswaran, L. and Lee, H. (2018).
An efficient framework for learning sentence representations.
In *In ICLR*.

[Martens and Sutskever, 2011] Martens, J. and Sutskever, I. (2011).
Learning recurrent neural networks with hessian-free optimization.
In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 1033–1040, USA. Omnipress.

[Mikolov et al., 2013] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013).
Distributed representations of words and phrases and their compositionality.
In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

[Olah and Carter, 2016] Olah, C. and Carter, S. (2016).
Attention and augmented recurrent neural networks.
Distill.

[Pascanu et al., 2013] Pascanu, R., Mikolov, T., and Bengio, Y. (2013).
On the difficulty of training recurrent neural networks.
In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pages III–1310–III–1318. JMLR.org.

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. D. (2014).
Glove: Global vectors for word representation.
In *In EMNLP*.

References IV

[Peters et al., 2018] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018).
Deep contextualized word representations.
In Walker, M. A., Ji, H., and Stent, A., editors, *NAACL-HLT*, pages 2227–2237. Association for Computational Linguistics.

[Siegelmann and Sontag, 1995] Siegelmann, H. and Sontag, E. (1995).
On the computational power of neural nets.
J. Comput. Syst. Sci., 50(1):132–150.

[Srivastava et al., 2015] Srivastava, R. K., Greff, K., and Schmidhuber, J. (2015).
Highway networks.
CoRR, abs/1505.00387.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017).
Attention is all you need.
In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

[Vinyals et al., 2015] Vinyals, O., Kaiser, L. u., Koo, T., Petrov, S., Sutskever, I., and Hinton, G. (2015).
Grammar as a foreign language.
In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28*, pages 2773–2781. Curran Associates, Inc.