TÉCNICO LISBOA

Intelligent Systems

Susana M. Vieira

Universidade de Lisboa, Instituto Superior Técnico

IS4, Center of Intelligent Systems, IDMEC, LAETA, Portugal
{susana.vieira}@tecnico.ulisboa.pt

Deep Learning

SI10 – Deep Learning – Text and Sequences

Reading:

- Ian Goodfellow. ***Deep Learning***. MIT Press, 2016.
- François Chollet. ***Deep Learning With Python***. 2nd Edition, 2017.
- S. Haykin. ***Neural Networks and Learning Machines***. Pearson Education, 2016.

Sequence Modeling: Recurrent and Recursive Nets

Natural Language Processing

- Sentence/Document level Classification (topic, sentiment)
- Topic modeling (LDA, ...)
- Translation
- Chatbots / dialogue systems / assistants (Alexa, ...)
- Summarization

Reading: *A Primer on Neural Network Models for Natural Language Processing* by Yoav Goldberg

Natural Language Processing

- Classification and word representation
- Word2Vec (embeddings)
- Language Modelling
- Recurrent neural networks

Embeddings

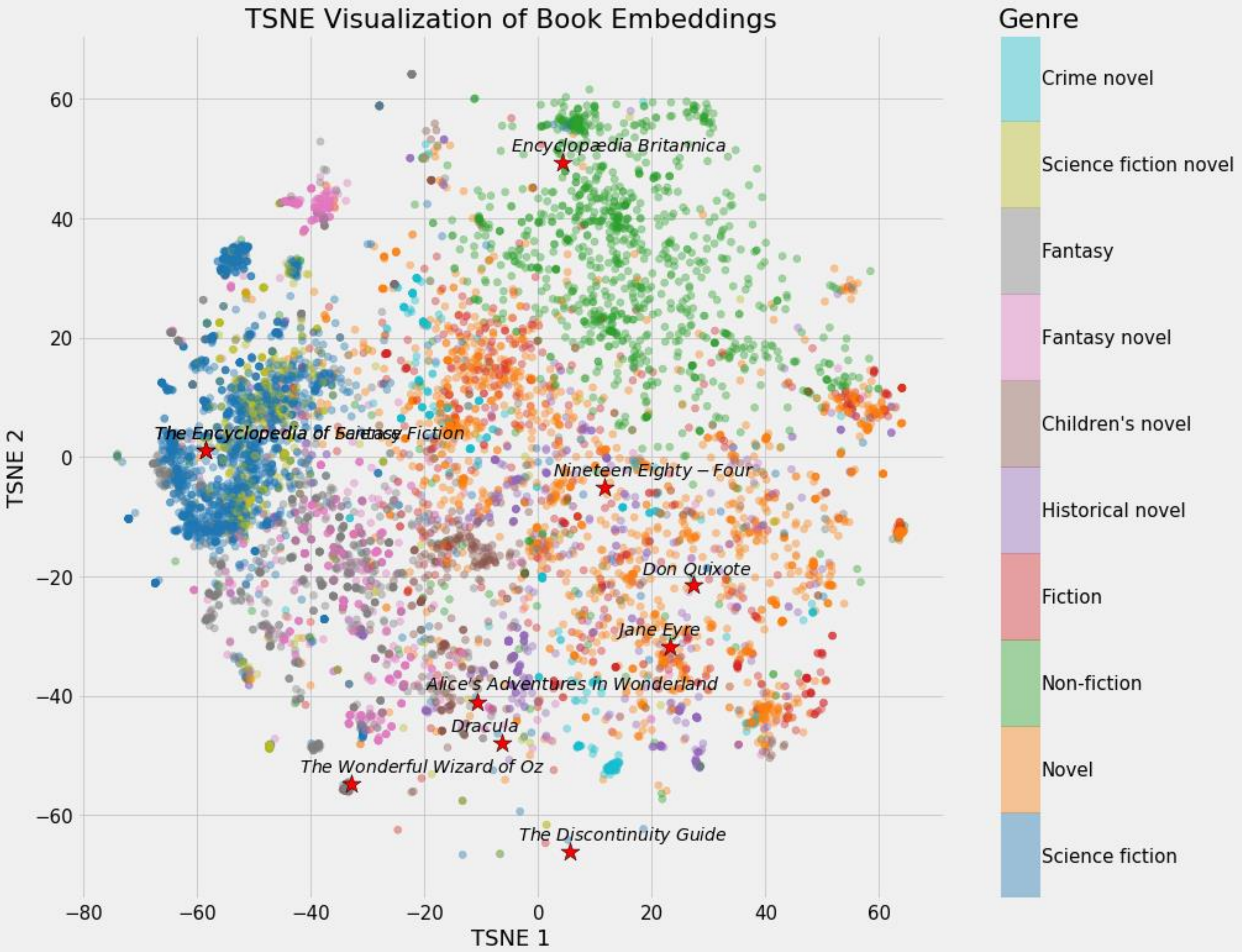
Embedding

- An embedding is a mapping of a discrete — categorical — variable to a vector of continuous numbers.
- In the context of neural networks, embeddings are low-dimensional, learned continuous vector representations of discrete variables.
- Neural network embeddings reduce the dimensionality of categorical variables and meaningfully represent categories in the transformed space.

Embedding

- Neural network embeddings have 3 primary purposes:
  - Finding nearest neighbors in the embedding space.

  (These can be used to make recommendations based on user interests or cluster categories)
  - As input to a machine learning model for a supervised task.
  - For visualization of concepts and relations between categories.



TÉCNICO LISBOA

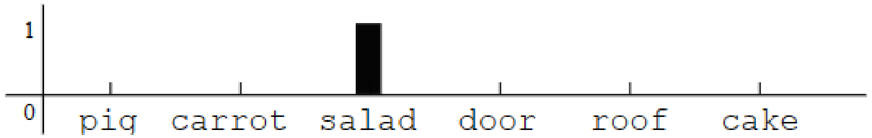
Symbolic Variables

- **Text**: characters, words, bigrams...
- **Recommender Systems**: item ids, user ids
- **Any categorical descriptor**: tags, movie genres, visited URLs, skills on a resume, product categories...
- Notation:

Symbol s in vocabulary V

One-hot representation

$$onehot(\text{'salad'}) = [0, 0, 1, \ldots, 0] \in \{0, 1\}^{|V|}$$



- Sparse, discrete, large dimension $|V|$
- Each axis has a meaning
- Symbols are equidistant from each other:

euclidean distance = $\sqrt{2}$

Embedding

$$embedding(\text{'salad'}) = [3.28, -0.45, \ldots 7.11] \in \mathbb{R}^d$$

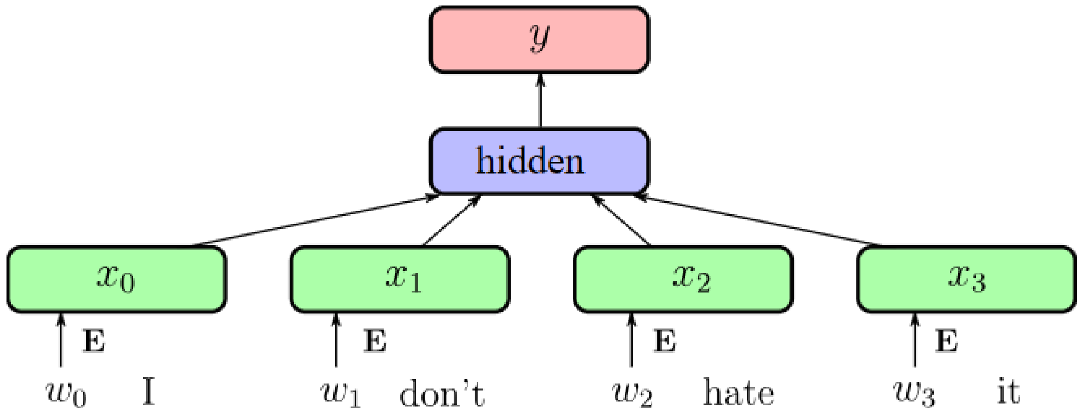
- Continuous and dense
- Can represent a huge vocabulary in low dimension, typically: $d \in \{16, 32, \ldots, 4096\}$
- Axis have no meaning a priori
- Embedding metric can capture semantic distance

Neural Networks compute transformations on continuous vectors

Word Representation

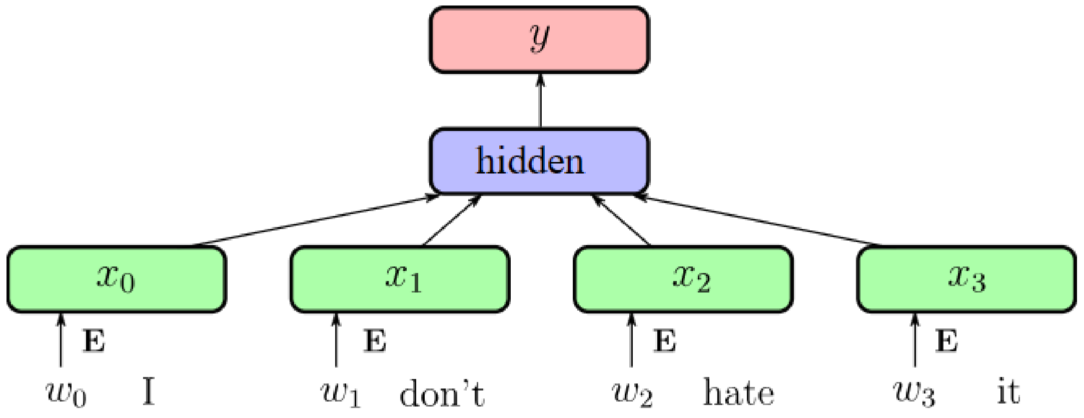
- Words are indexed and represented as 1-hot vectors
- Large Vocabulary of possible words $|V|$
- Use of Embeddings as inputs in all Deep NLP tasks
- Word embeddings usually have dimensions 50, 100, 200, 300

Supervised Text Classification



- $\mathbf{E}$ embedding (linear projection) — $|V| \times H$
- Embeddings are averaged — hidden activation size: H
- Dense output connection $\mathbf{W},\mathbf{b}$ — $H \times K$
- Softmax and **cross-entropy** loss

Supervised Text Classification



- Very efficient (**speed** and **accuracy**) on large datasets
- State-of-the-art (or close to) on several classification, when adding **bigrams/trigrams**
- Little gains from depth

Reading: Joulin, Armand, et al. "Bag of tricks for efficient text classification." FAIR 2016

Transfer Learning for Text

- Similar to image: can we have word representations that are generic enough to **transfer** from one task to another?
- **Unsupervised** / **self-supervised** learning of word representations
- **Unlabeled** text data is almost infinite:
  - Wikipedia dumps
  - Project Gutenberg
  - Social Networks
  - Common Crawl

Word to Vectors

Excerpt from work by J. Turian on a model trained by R. Collobert et al. 2008

Word2Vec

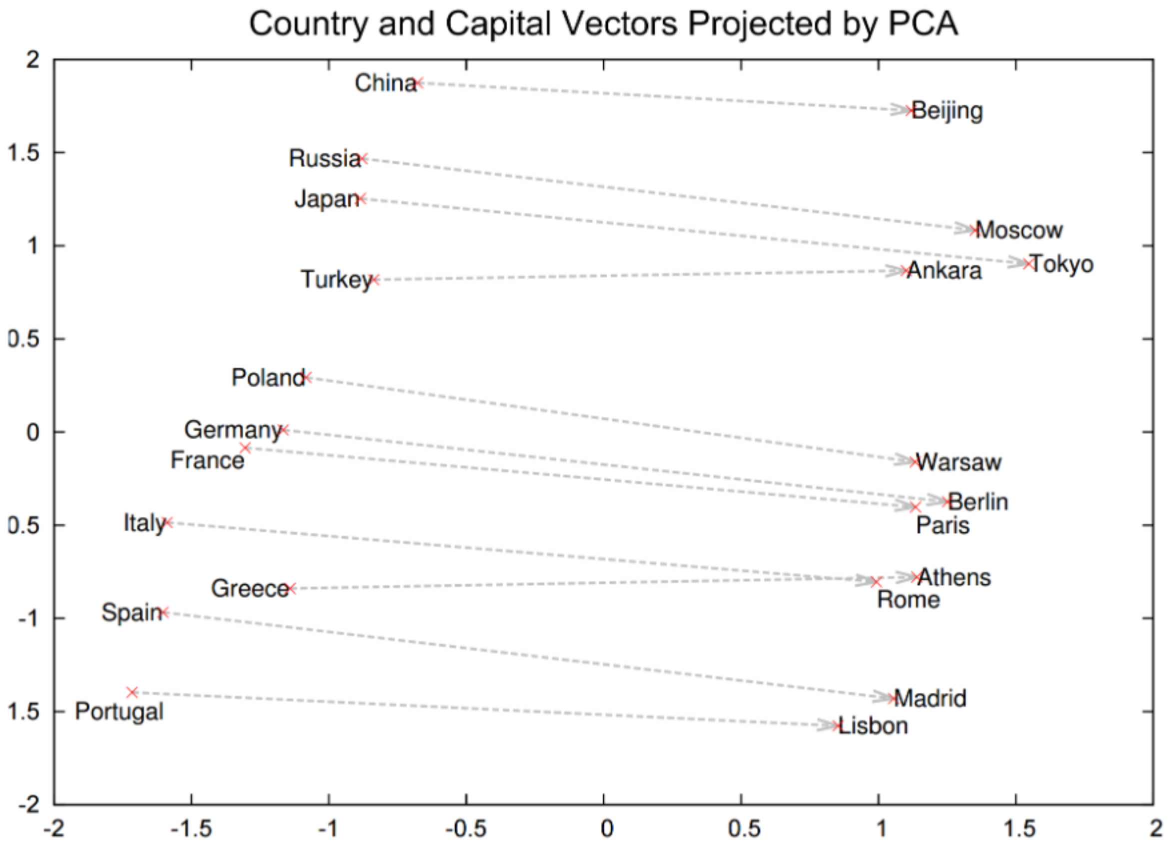
FRANCE	JESUS	XBOX	REDDISH	SCRATCHED	MEGABITS
AUSTRIA	GOD	AMIGA	GREENISH	NAILED	OCTETS
BELGIUM	SATI	PLAYSTATION	BLUISH	SMASHED	MB/S
GERMANY	CHRIST	MSX	PINKISH	PUNCHED	BIT/S
ITALY	SATAN	IPOD	PURPLISH	POPPED	BAUD
GREECE	KALI	SEGA	BROWNISH	CRIMPED	CARATS
SWEDEN	INDRA	PSNUMBER	GREYISH	SCRAPED	KBIT/S
NORWAY	VISHNU	HD	GRAYISH	SCREWED	MEGAHERTZ
EUROPE	ANANDA	DREAMCAST	WHITISH	SECTIONED	MEGAPIXELS
HUNGARY	PARVATI	GEFORCE	SILVERY	SLASHED	GBIT/S
SWITZERLAND	GRACE	CAPCOM	YELLOWISH	RIPPED	AMPERES

Compositionality

Czech + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
koruna	Hanoi	airline Lufthansa	Moscow	Juliette Binoche
Check crown	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile De

Colobert et al. 2011, Mikolov, et al. 2013

Word Analogies



- Linear relations in Word2Vec embeddings
- Many come from text structure (e.g. Wikipedia)

Reading: Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." NIPS 2013

TÉCNICO LISBOA

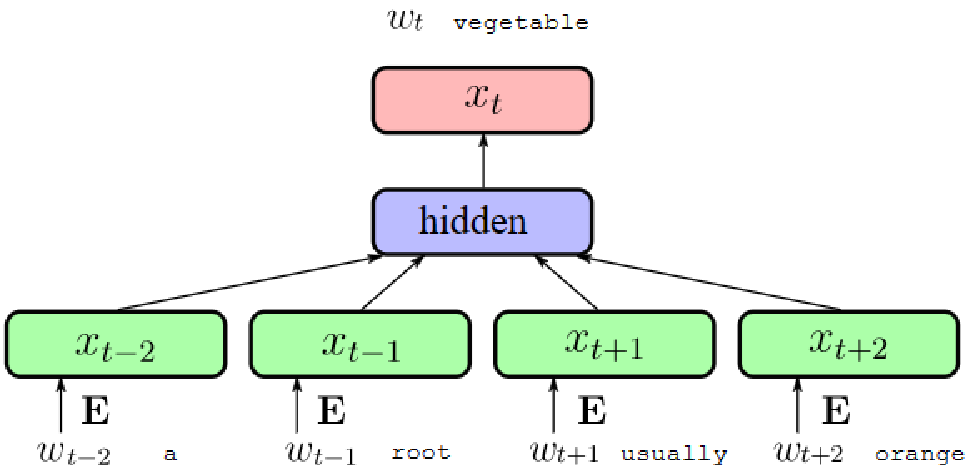
Self-supervised Training

- Distributional Hypothesis (Harris, 1954): "*words are characterised by the company that they keep*"
- Main idea: learning word embeddings by **predicting word contexts**
- Given a word e.g. "carrot" and any other word $w \in V$ predict probability $P(w|\text{carrot})$ that w occurs in the context of "carrot".
  - **Unsupervised / self-supervised**: no need for class labels.
  - (Self-)supervision comes from **context**.
  - Requires a lot of text data to cover rare words correctly.

Word2Vec:CBoW

- CBoW: representing the context as **Continuous Bag-of-Word**
- Self-supervision from large unlabeled corpus of text: *slide* over an **anchor word** and its **context**:

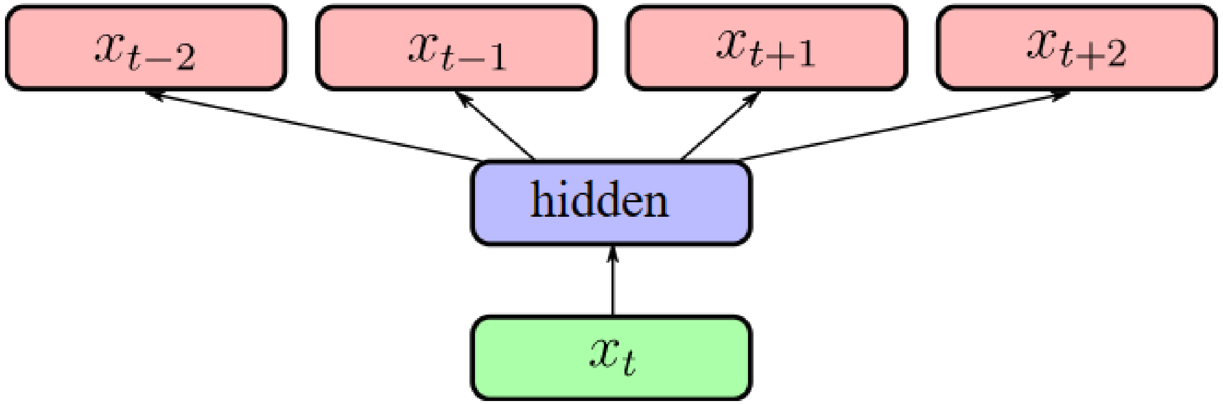
`the carrot is a root vegetable, usually orange`



Reading: Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." NIPS 2013

TÉCNICO LISBOA

Word2Vec: SkipGram



- Given the central word, predict occurence of other words in its context.
- Widely used in practice.
- Use **Negative Sampling**: sample negative words at random instead of computing the full softmax.

Reading: Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." NIPS 2013

Evaluation and Related Methods

- Always difficult to evaluate unsupervised tasks
  - WordSim (Finkelstein et al.)
  - SimLex-999 (Hill et al.)
  - Word Analogies (Mikolov et al.)

- Other popular method: GloVe (Socher et al.)
  http://nlp.stanford.edu/projects/glove/

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global Vectors for Word Representation." EMNLP. 2014

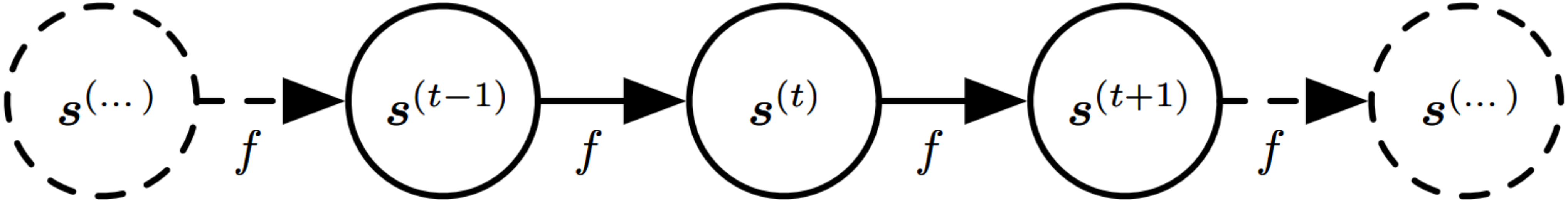
Take away on embeddings

For text applications, inputs of Neural Networks are Embeddings

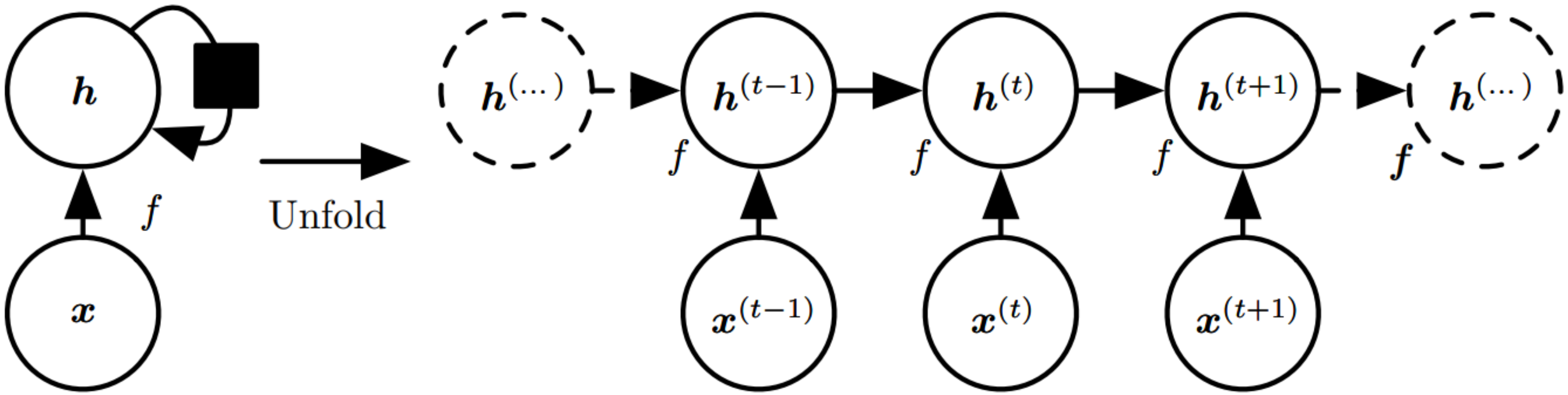
- If little training data and a wide vocabulary not well covered by training data, use pre-trained self-supervised embeddings (transfer learning from Glove, word2vec or fastText embeddings)
- If large training data with labels, directly learn task-specific embedding with methods such as fastText in supervised mode.
- These methods use Bag-of-Words (BoW): they ignore the order in word sequences
- Depth & non-linear activations on hidden layers are not that useful for BoW text classification.

Word Embeddings no long state of the art for NLP tasks: BERT-style pretraining of deep transformers with sub-word tokenization is now used everywhere.

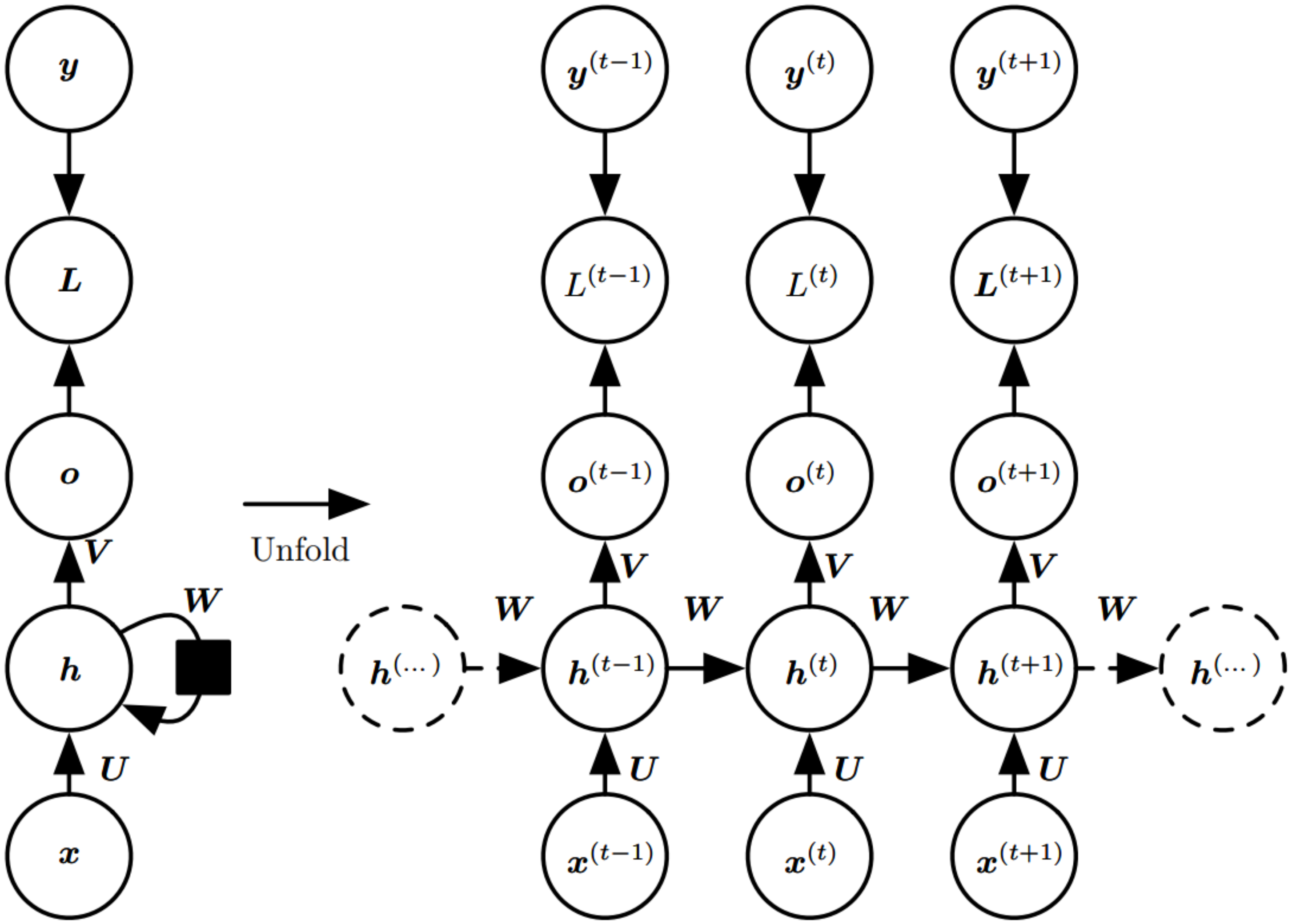
Classic Dynamic Systems



Unfolding Computation Graphs



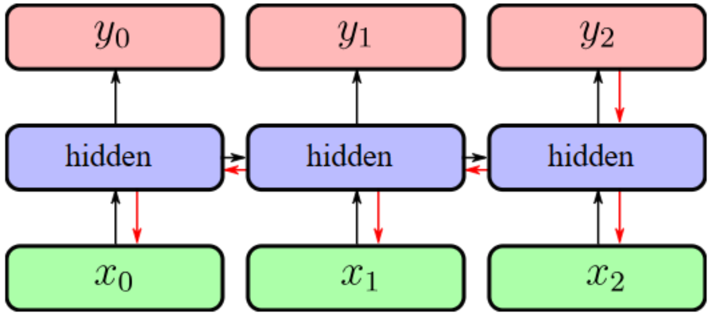
Recurrent Hidden Units



TÉCNICO LISBOA

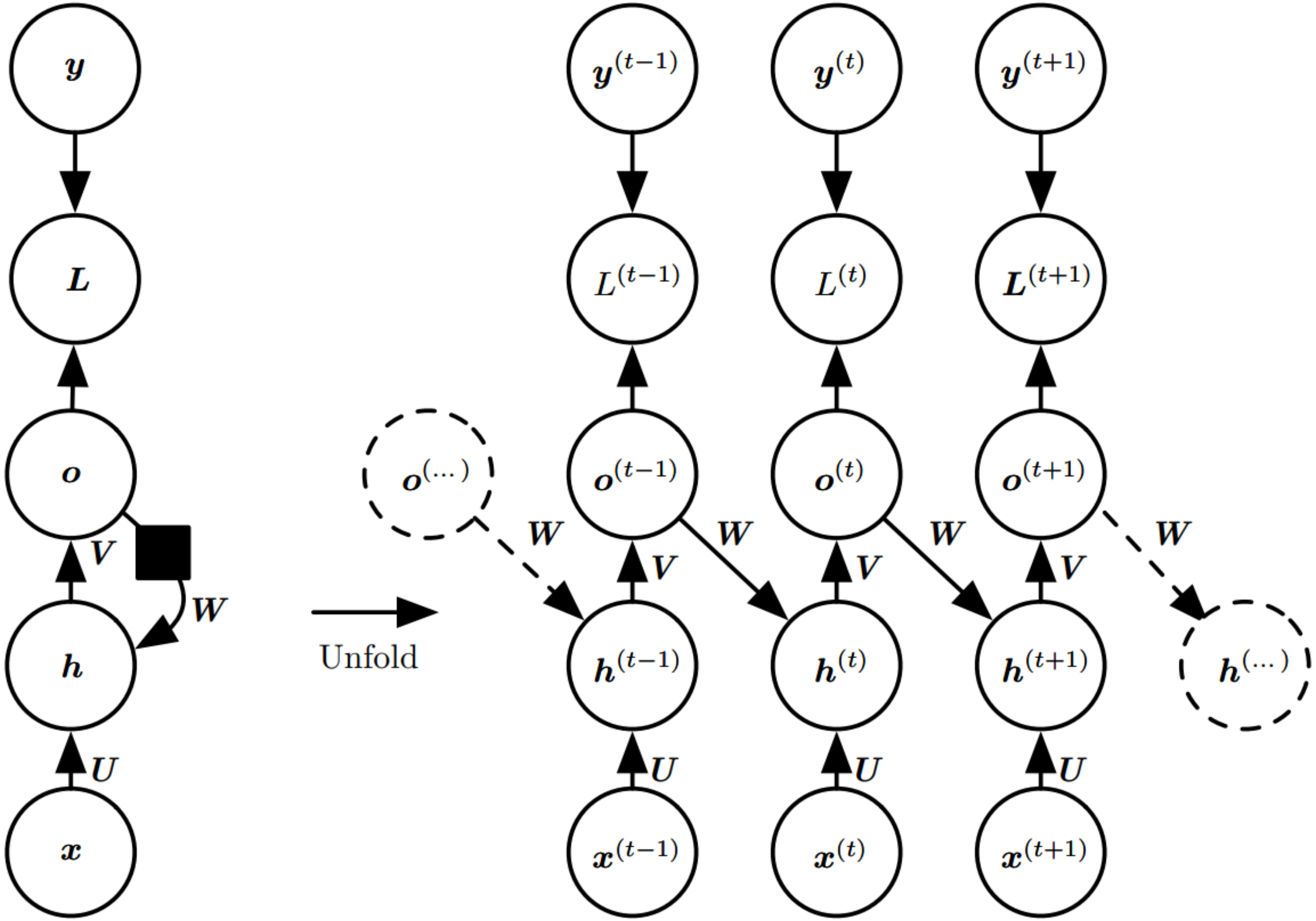
Backpropagation Through Time

- Similar as standard backpropagation on unfolded network:

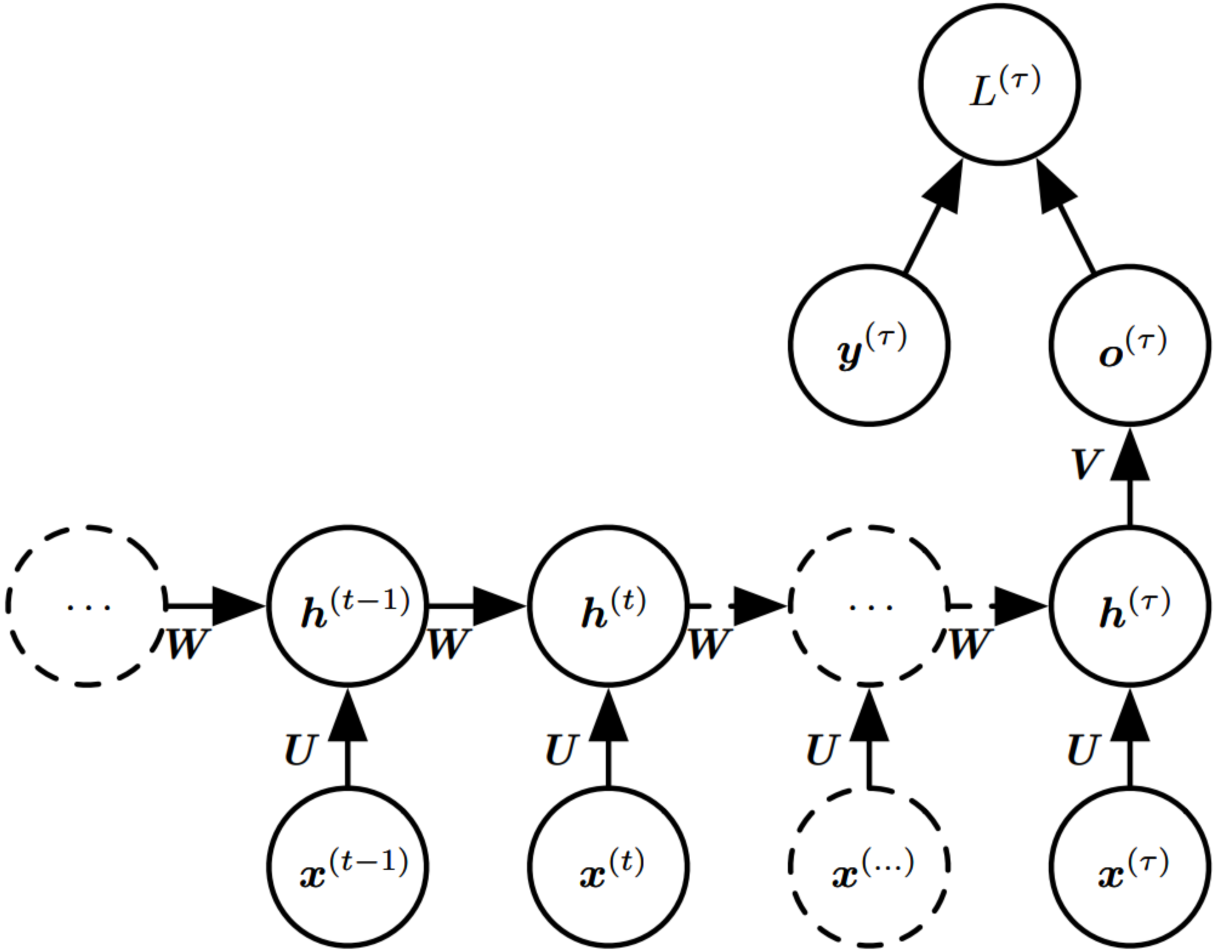


- Similar as training **very deep networks** with tied parameters
- Example between x_0 and y_2: W^h is used twice
- Usually truncate the backprop after T timesteps
- Difficulties to train long-term dependencies

Recurrence through only the Output



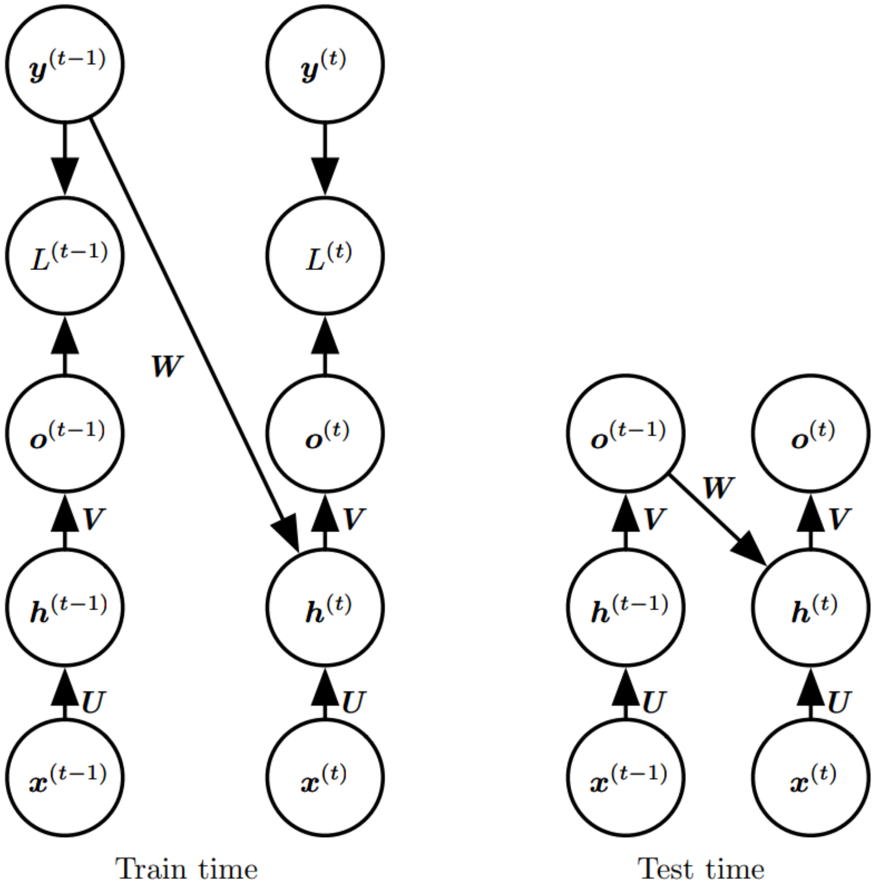
Sequence Input, Single Output



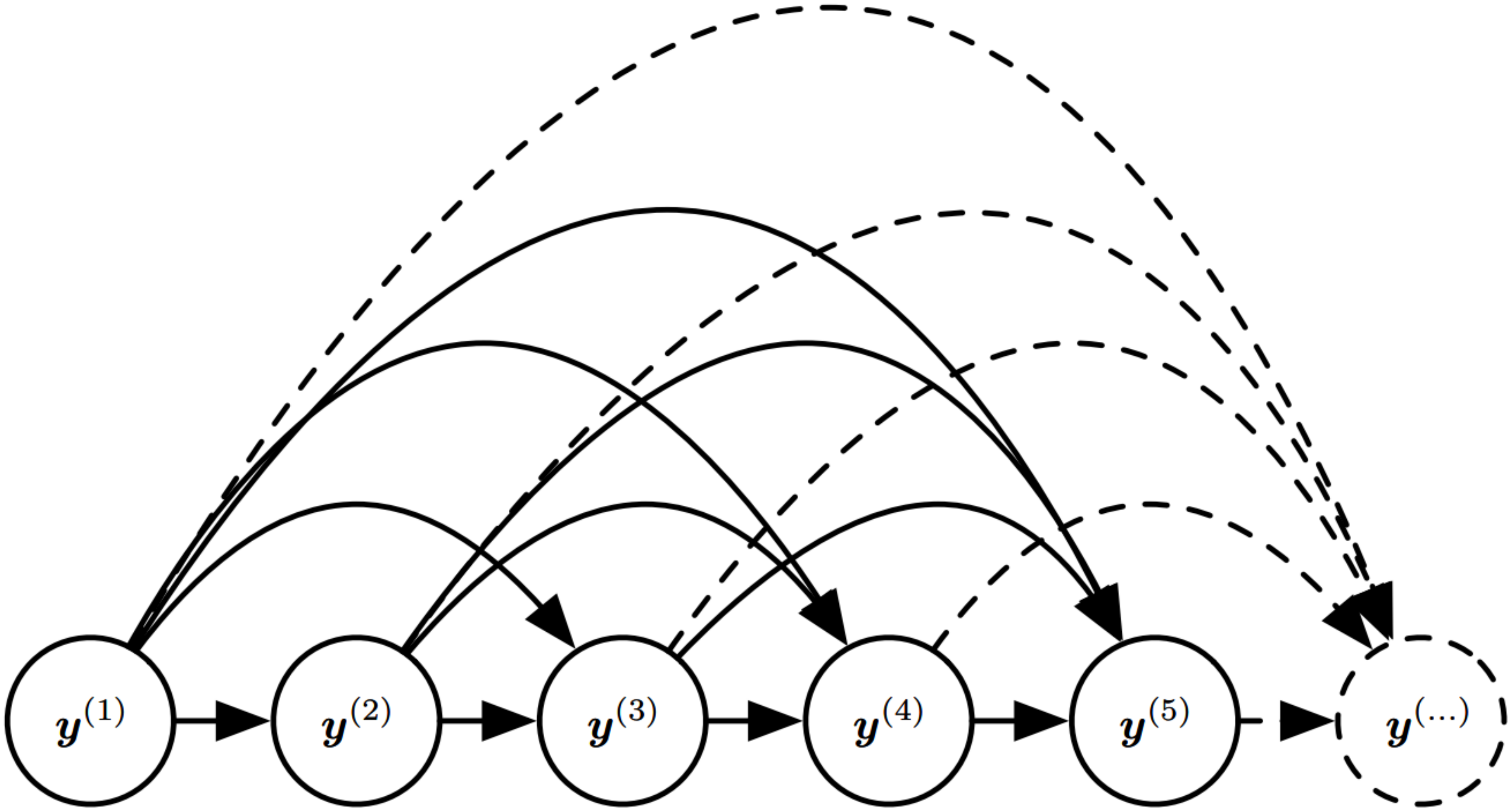
Used in **Sentiment Analysis**

Teacher Forcing

- ***"Teacher forcing"*** was introduced in 1989 by *Ronald J. Williams and David Zipser*

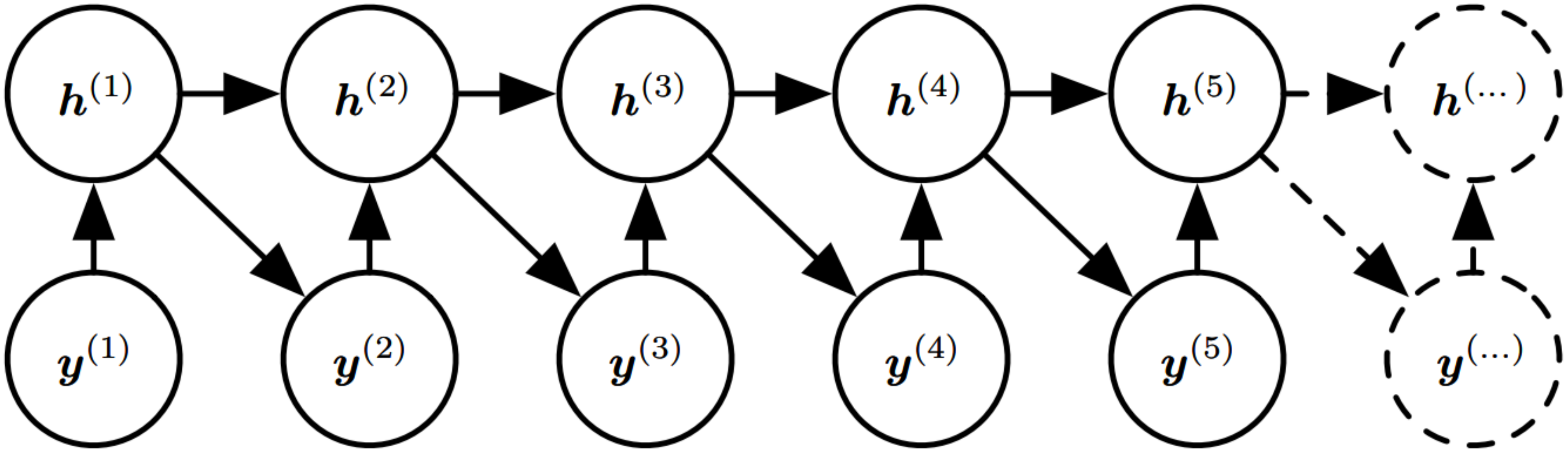


Fully Connected Graphical Model



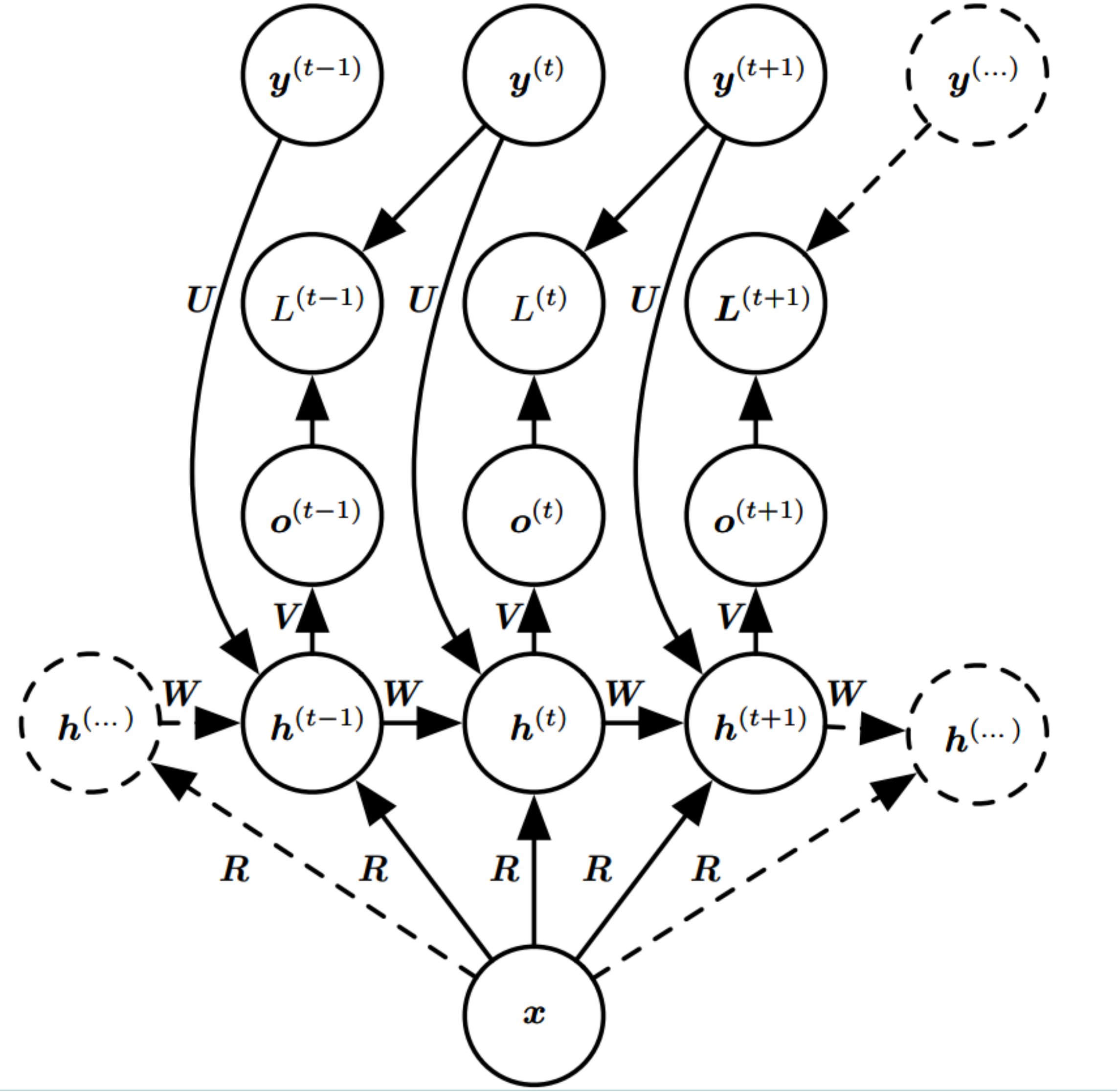
TÉCNICO LISBOA

RNN Graphical Model

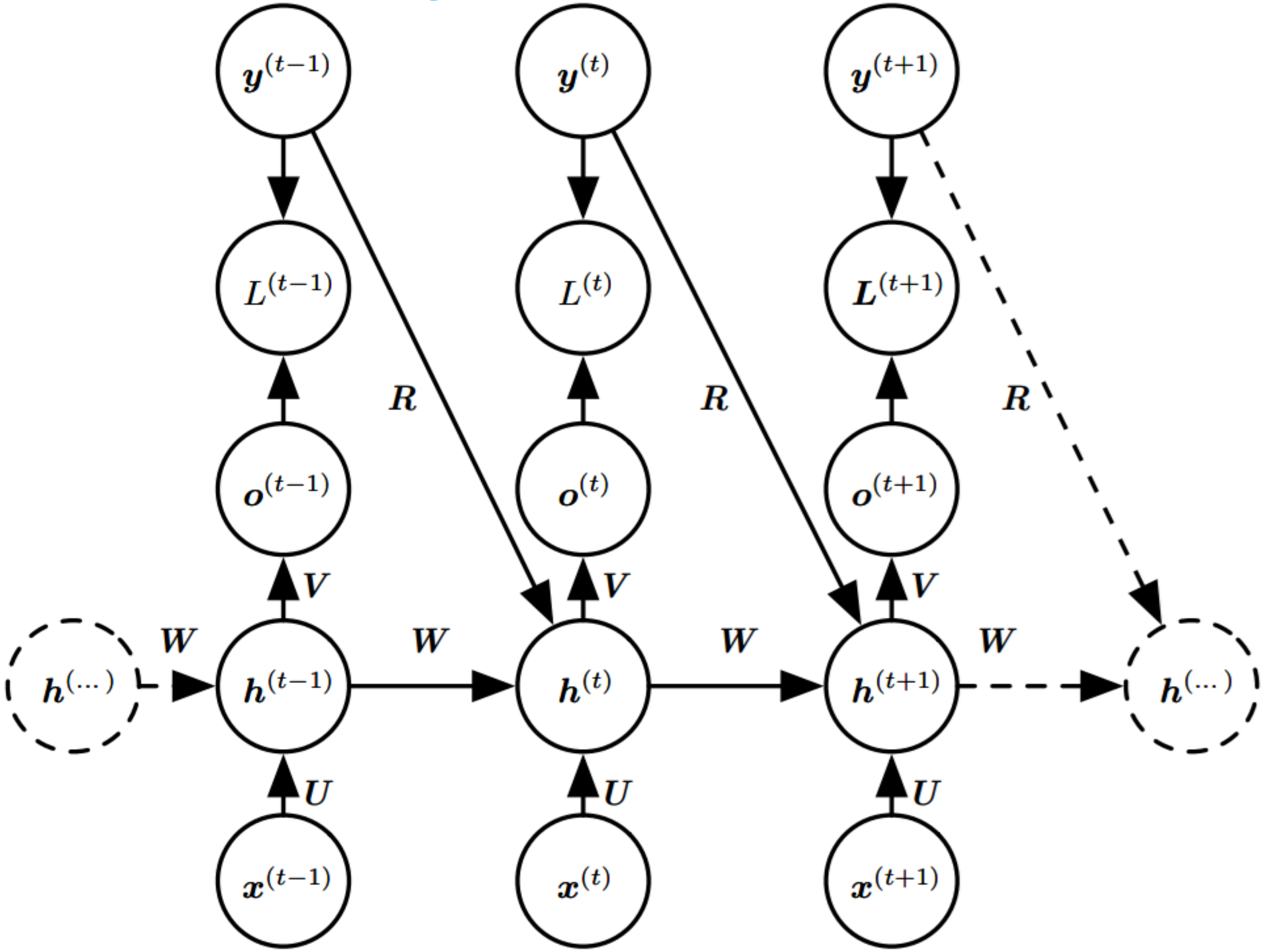


The conditional distributions for the hidden units are deterministic

Vector to Sequence

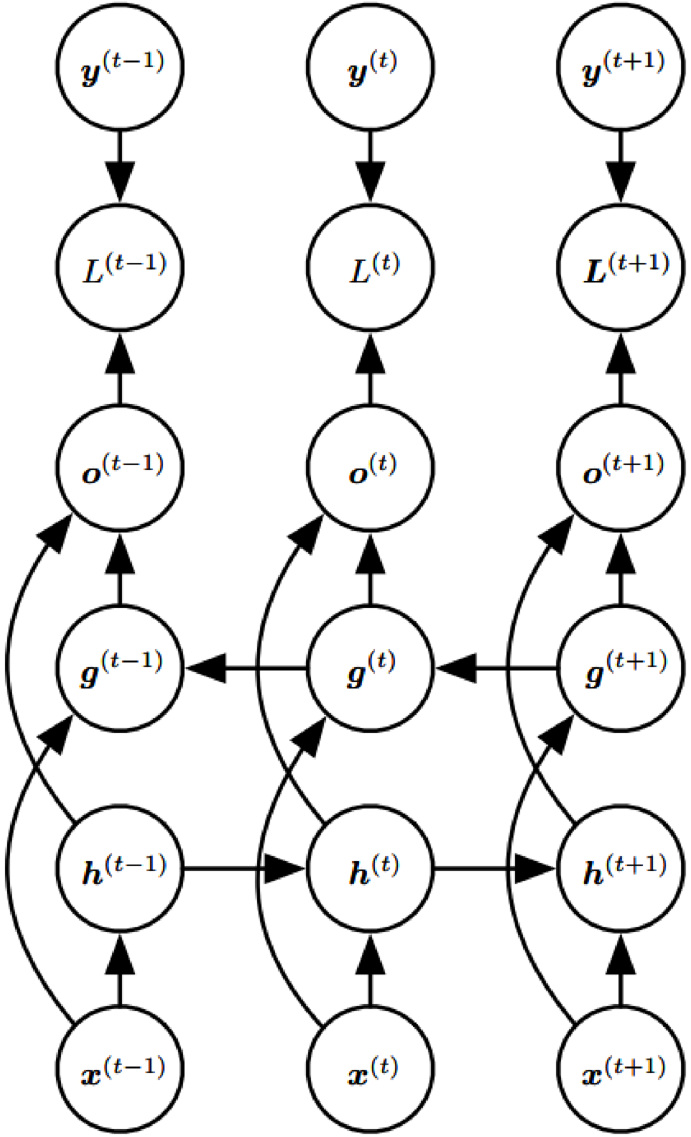


Hidden and Output Recurrence

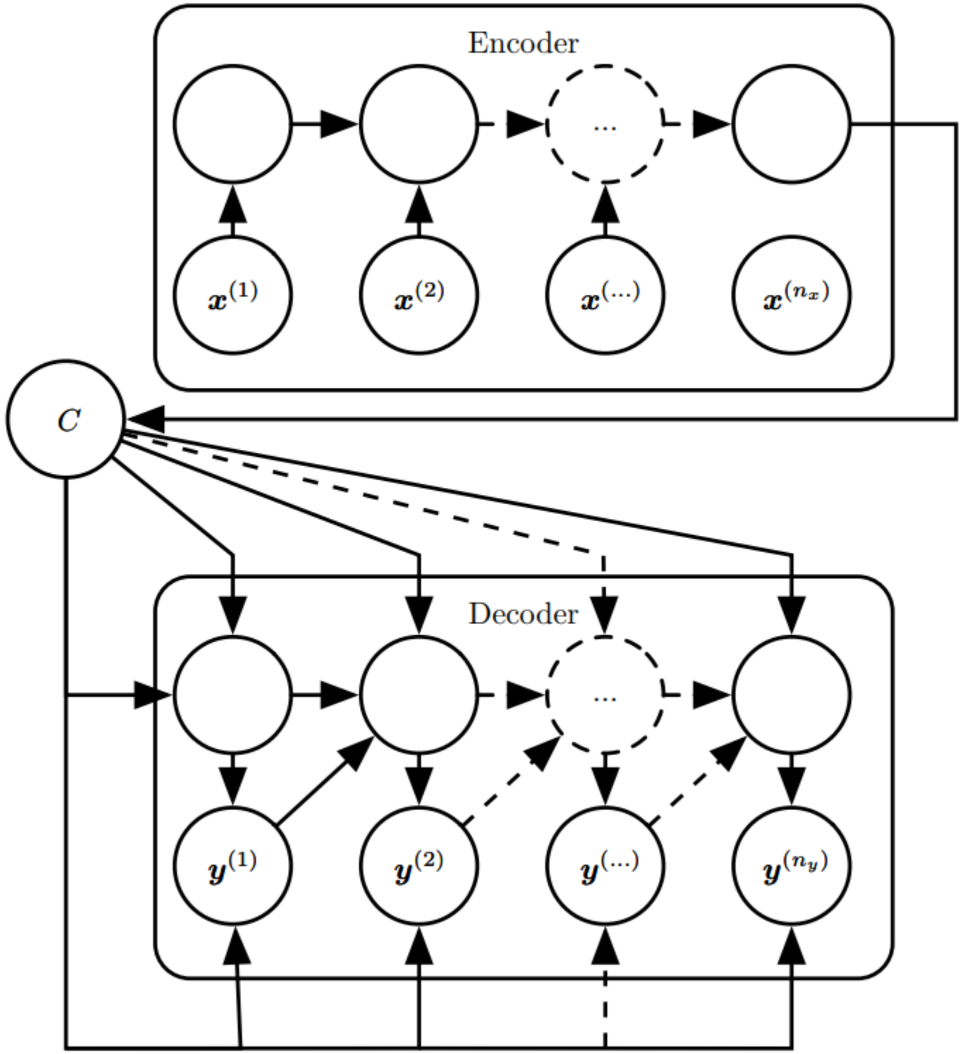


The output values are not forced to be conditionally independent in this model

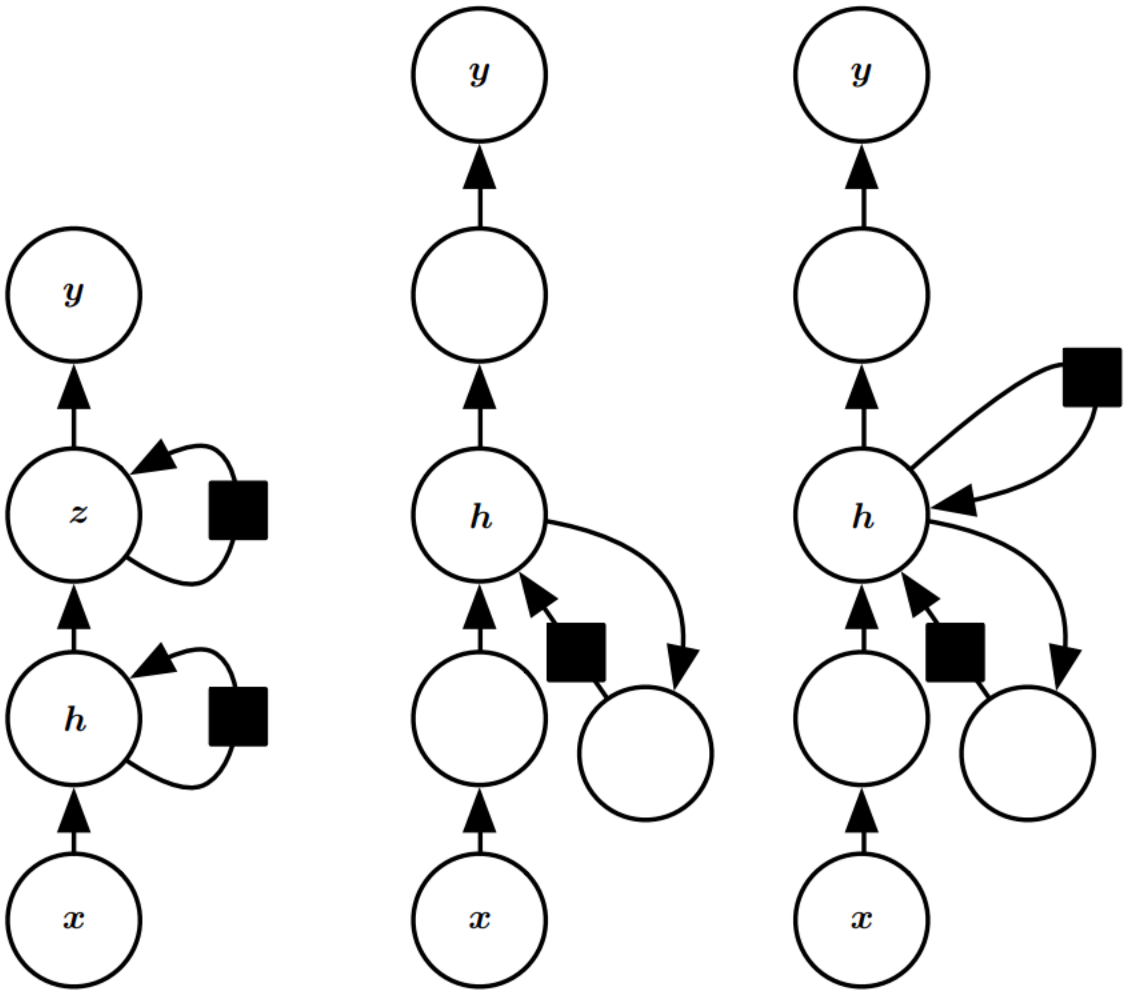
Bidirectional RNN



Sequence to Sequence Architecture

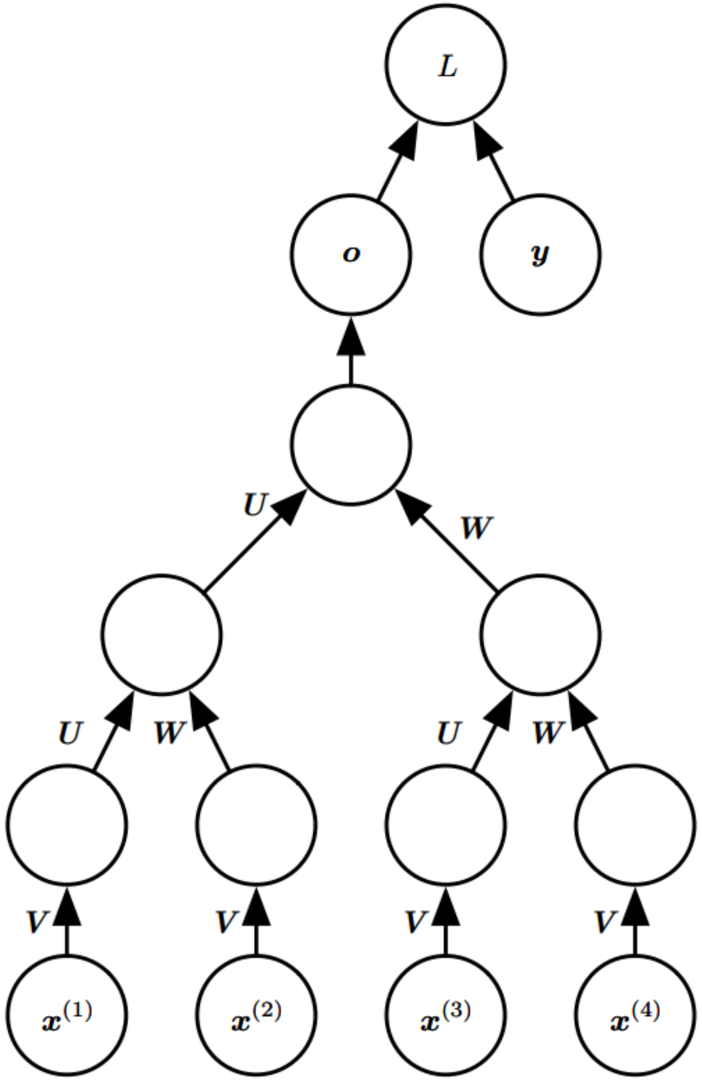


Deep RNNs

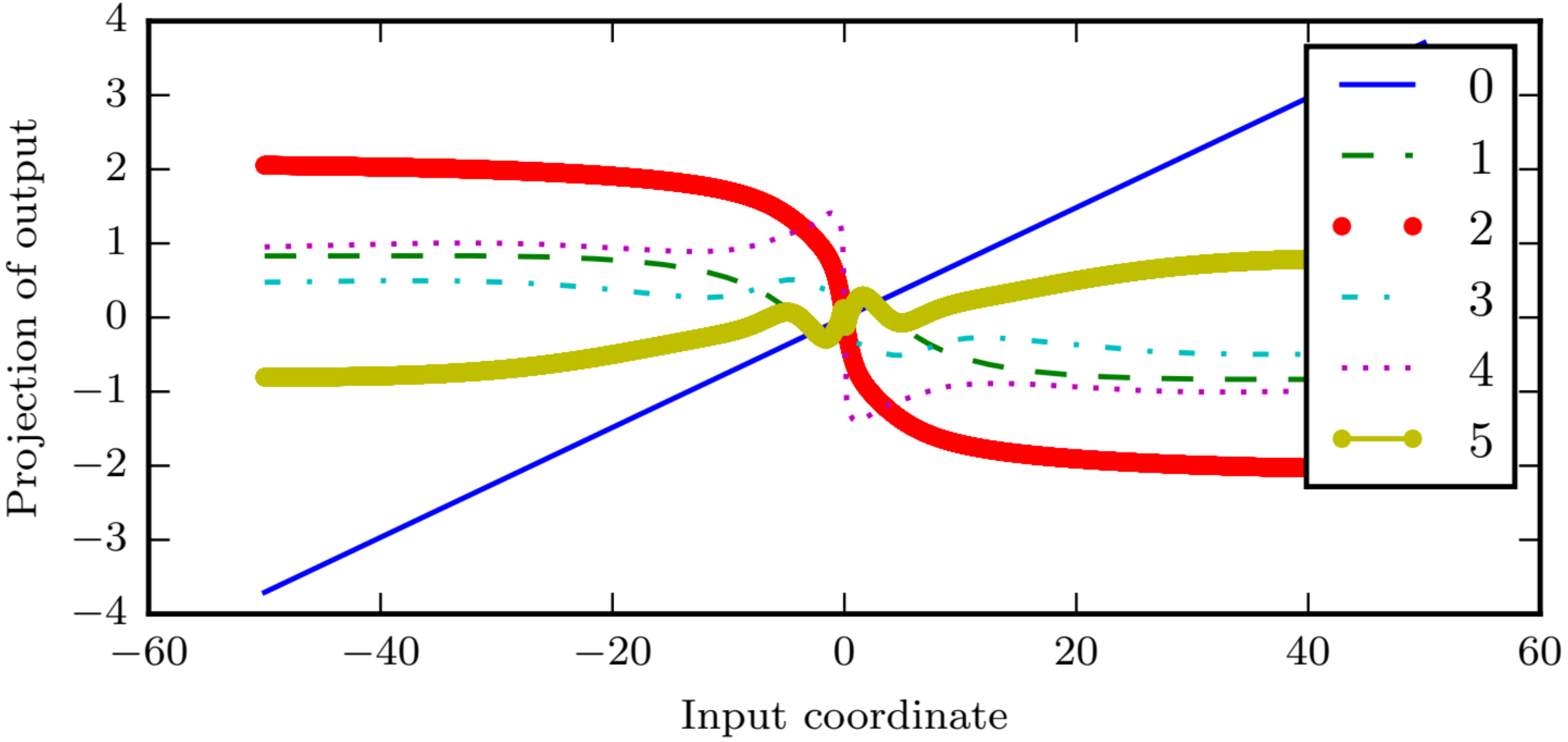


(a) (b) (c)

Recursive Network

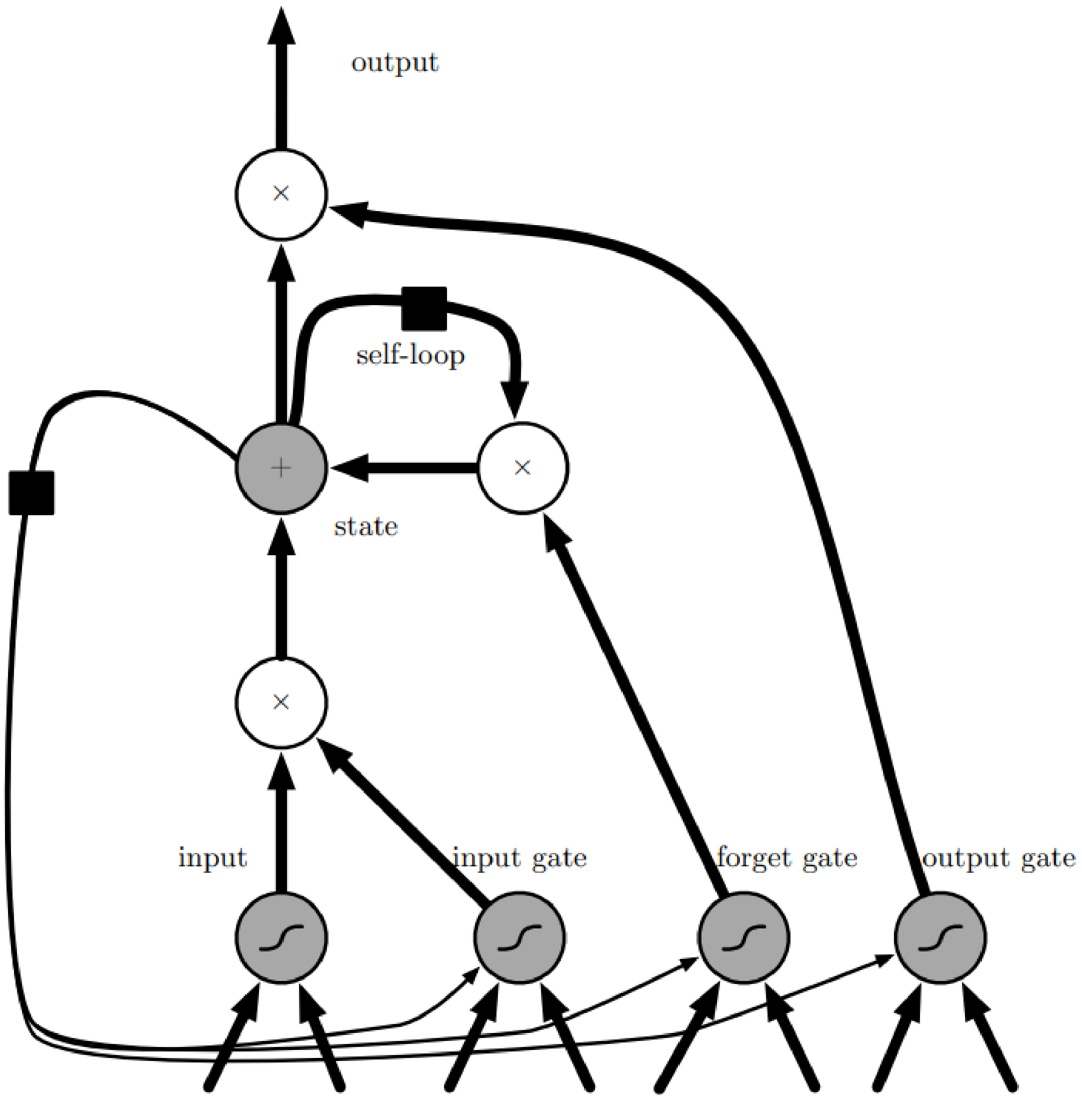


Exploding Gradients from Function Composition



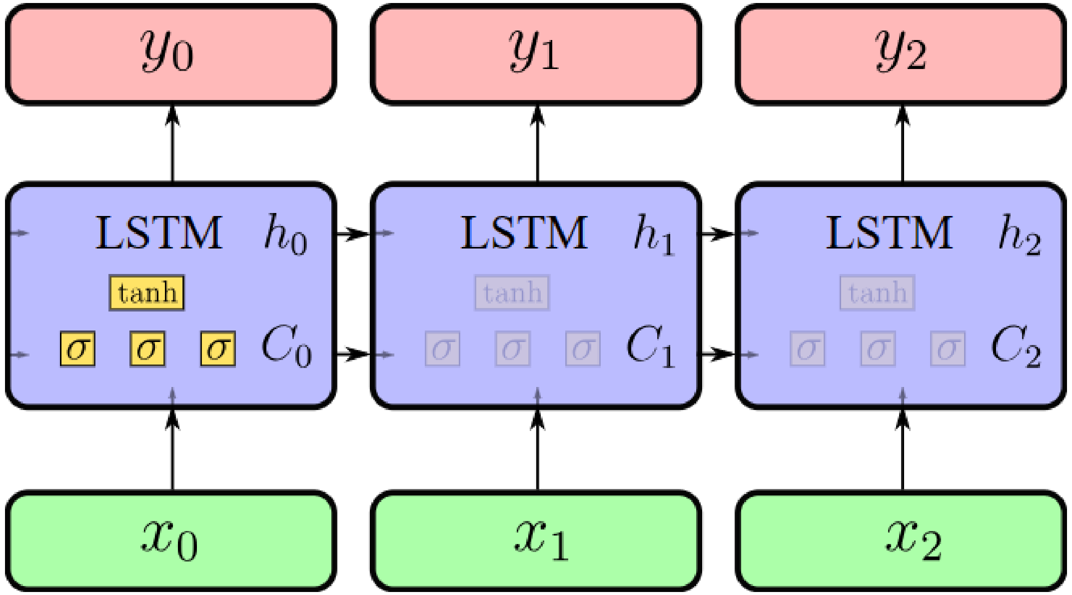
TÉCNICO LISBOA

LSTM



Long Short Term Memory

LSTM



- 4 times more parameters than RNN
- Mitigates **vanishing gradient** problem through **gating**
- Widely used and SOTA in many sequence learning problems

Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 1997

LSTMs

$\mathbf{u} = \sigma(\mathbf{W^u} \cdot h_{t-1} + \mathbf{I^u} \cdot x_t + b^u)$	Update gate H
$\mathbf{f} = \sigma(\mathbf{W^f} \cdot h_{t-1} + \mathbf{I^f} \cdot x_t + b^f)$	Forget gate H
$\tilde{\mathbf{c_t}} = \tanh(\mathbf{W^c} \cdot h_{t-1} + \mathbf{I^c} \cdot x_t + b^c)$	Cell candidate H
$\mathbf{c_t} = \mathbf{f} \odot \mathbf{c_{t-1}} + \mathbf{u} \odot \tilde{\mathbf{c_t}}$	Cell output H
$\mathbf{o} = \sigma(\mathbf{W^o} \cdot h_{t-1} + \mathbf{I^o} \cdot x_t + b^o)$	Output gate H
$\mathbf{h_t} = \mathbf{o} \odot \tanh(\mathbf{c_t})$	Hidden output H
$y = \mathrm{softmax}(\mathbf{W} \cdot h_t + b)$	Output K
W^u, W^f, W^c, W^o	Recurrent weights H x H
I^u, I^f, I^c, I^o	Input weights N x H

Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 1997

TÉCNICO LISBOA

GRU: Gate Recurrent Units

Gated Recurrent Unit: similar idea as LSTM

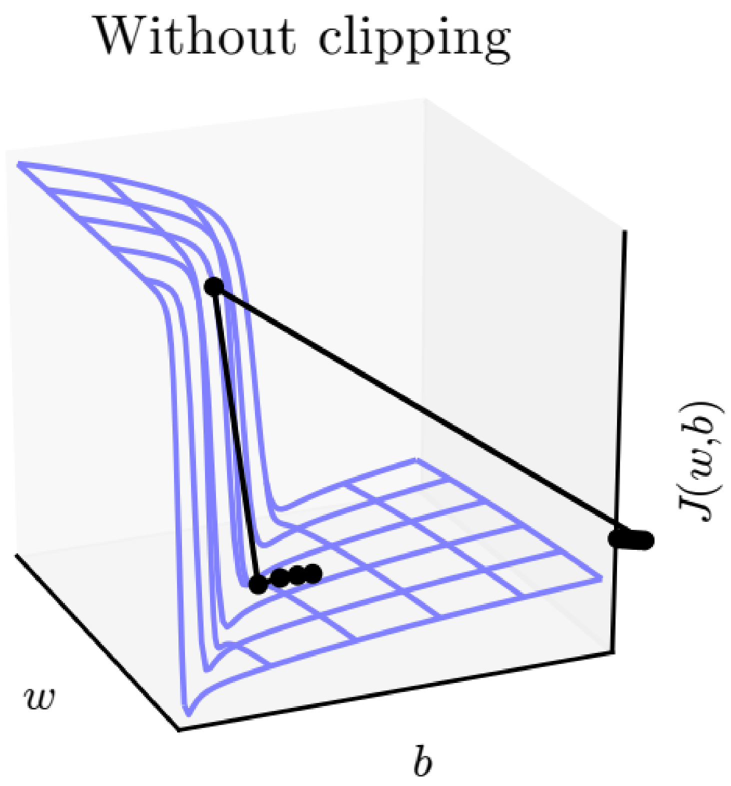
- less parameters, as there is one gate less
- no "cell", only hidden vector h_t is passed to next unit

In practice

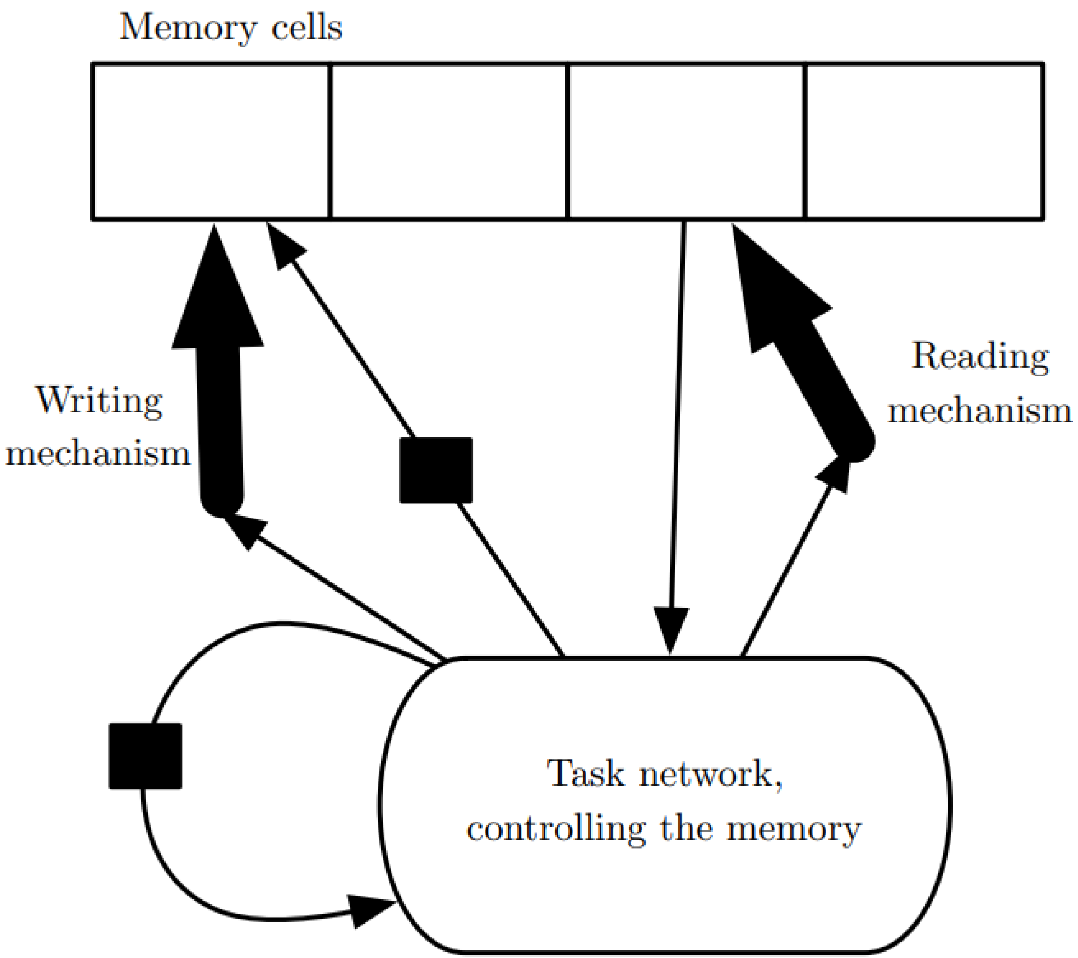
- more recent, people tend to use LSTM more
- no systematic difference between the two

Chung, Junyoung, et al. "Gated Feedback Recurrent Neural Networks." ICML 2015

Gradient Clipping

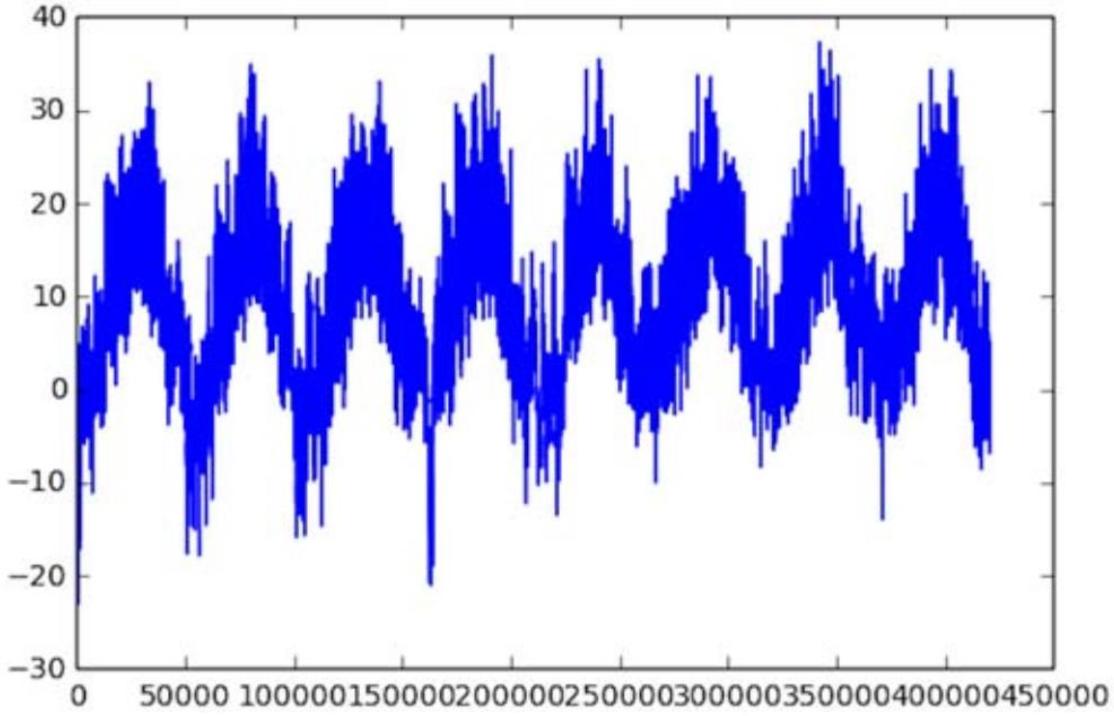


Networks with Explicit Memory



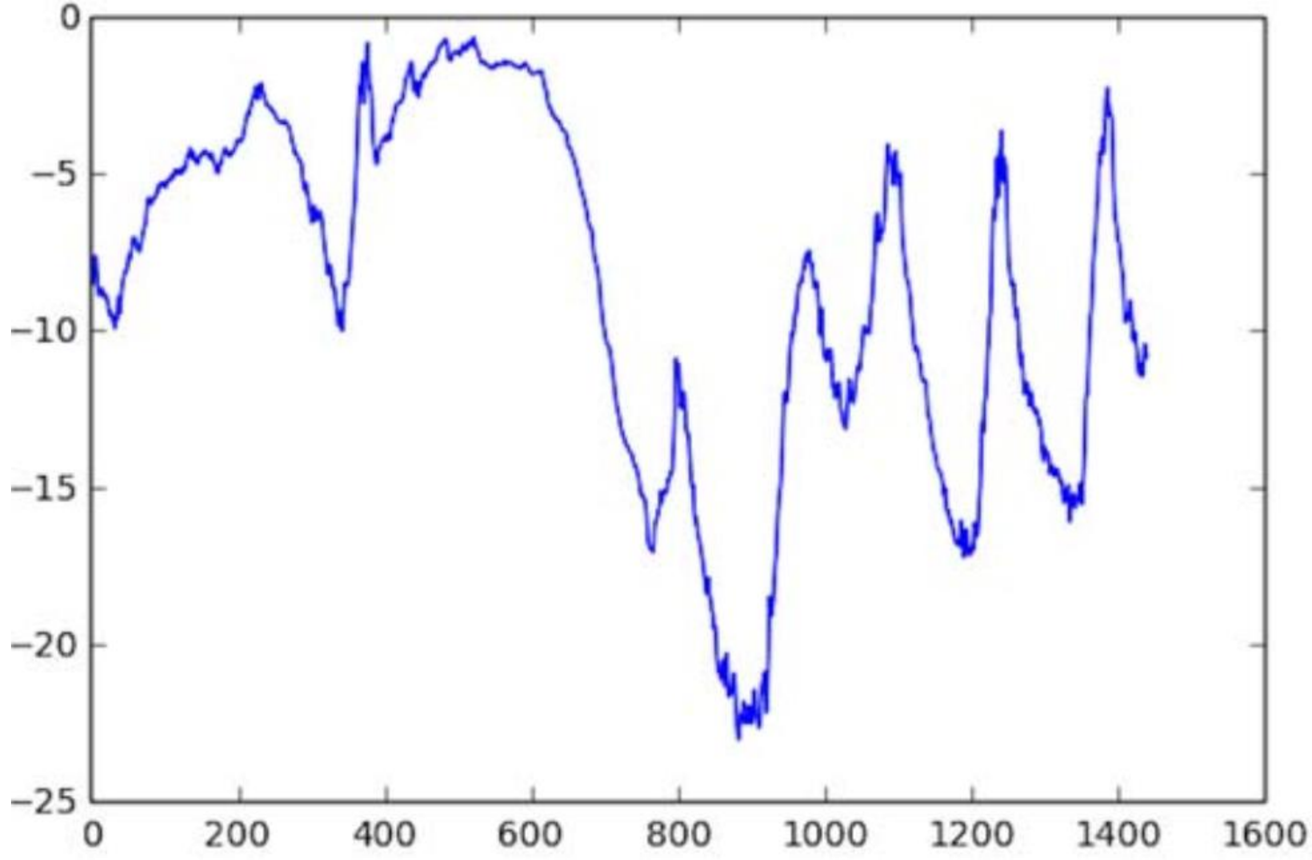
A temperature Forecasting Problem

- Weather timeseries dataset: www.bgc-jena.mpg.de/wetter
- Use data from 2009–2016: Temperature (°C)



A temperature Forecasting Problem

- Temperature (°C) of the first 10 days:



TÉCNICO LISBOA

Preparing the data

- Given data going as far back as **lookback** timesteps (a timestep is 10 minutes) and sampled every **steps** timesteps, can you predict the temperature in **delay** timesteps?
  - `lookback = 720`—Observations will go back 5 days.
  - `steps = 6`—Observations will be sampled at one data point per hour.
  - `delay = 144`—Targets will be 24 hours in the future.

Normalizing the data

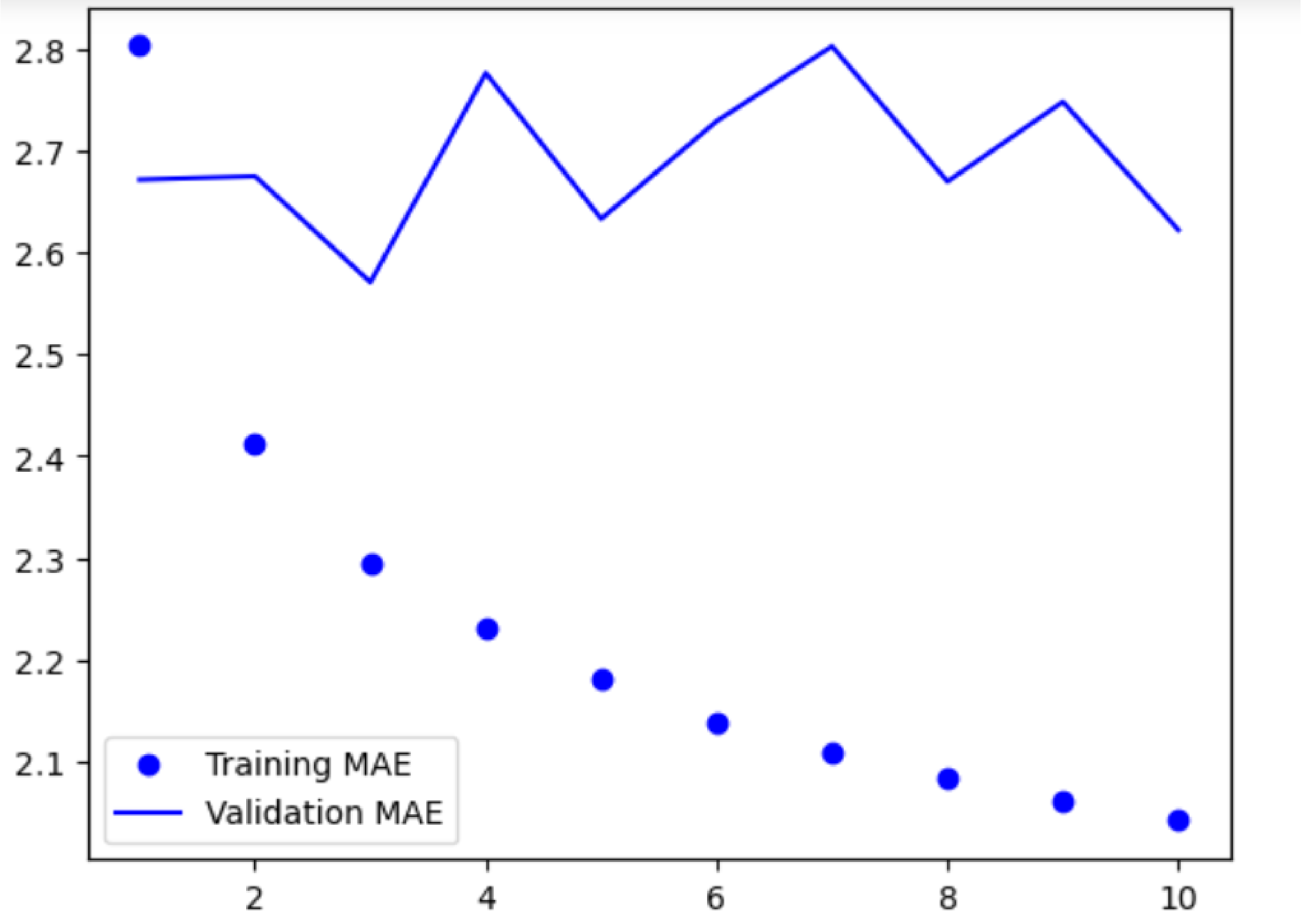
- Normalize each timeseries independently so that they all take small values on a similar scale.
- Preprocess the data by subtracting the mean of each timeseries and dividing by the standard deviation.
- Use first 200,000 timesteps as training data: compute the mean and standard deviation on this fraction of the data.

Training and evaluating a densely connected model

```
inputs = keras.Input(shape=(sequence_length, raw_data.shape[-1]))
x = layers.Flatten()(inputs)
x = layers.Dense(16, activation="relu")(x)
outputs = layers.Dense(1)(x)
model = keras.Model(inputs, outputs)

callbacks = [
    keras.callbacks.ModelCheckpoint("jena_dense.keras",
                                    save_best_only=True)
]
model.compile(optimizer="rmsprop", loss="mse", metrics=["mae"])
history = model.fit(train_dataset,
                    epochs=10,
                    validation_data=val_dataset,
                    callbacks=callbacks)
```

Training and evaluating a densely connected model

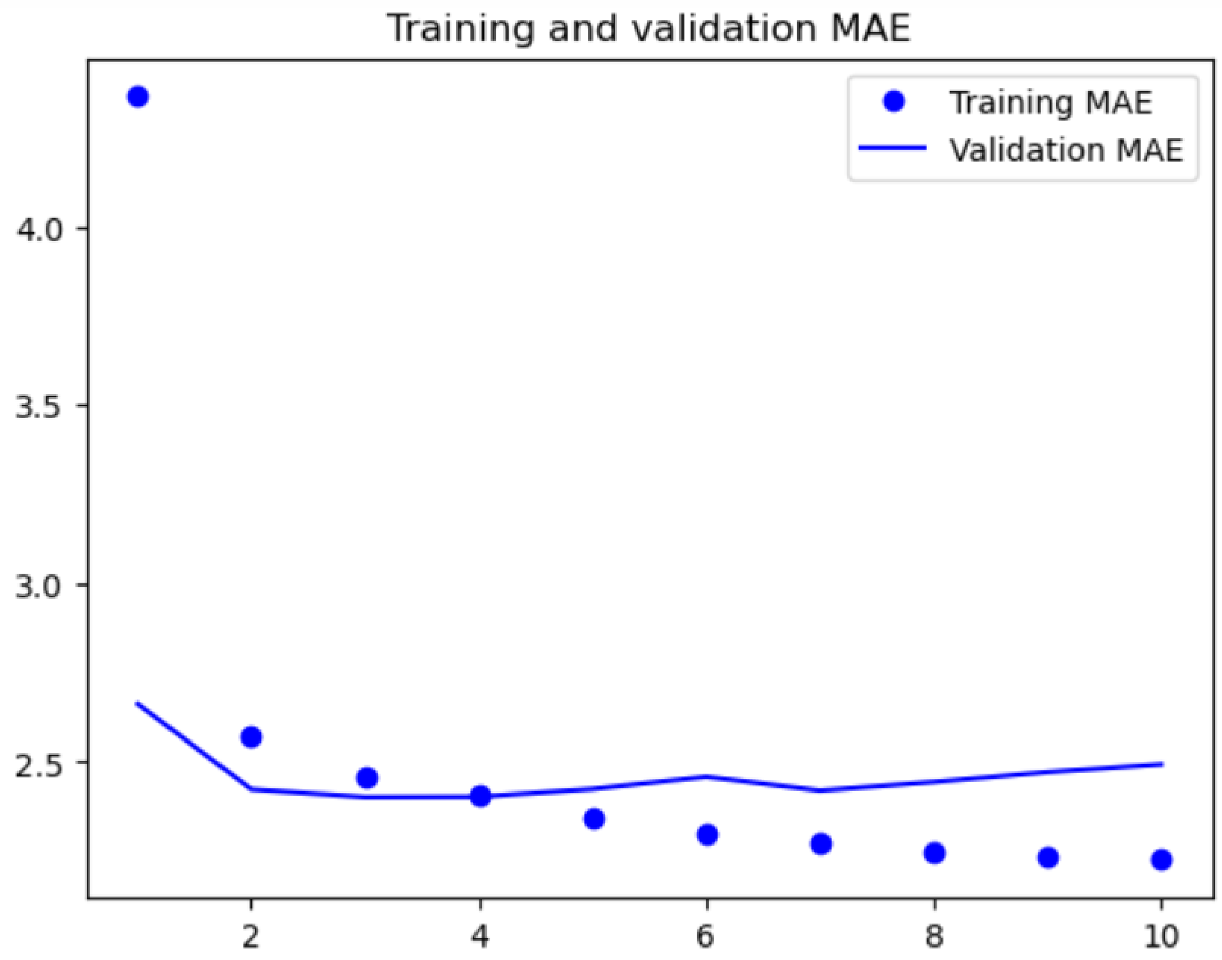


Training and evaluating a simple LSTM-based model

```
inputs = keras.Input(shape=(sequence_length, raw_data.shape[-1]))
x = layers.LSTM(16)(inputs)
outputs = layers.Dense(1)(x)
model = keras.Model(inputs, outputs)

callbacks = [
    keras.callbacks.ModelCheckpoint("jena_lstm.keras",
                                    save_best_only=True)
]
model.compile(optimizer="rmsprop", loss="mse", metrics=["mae"])
history = model.fit(train_dataset,
                    epochs=10,
                    validation_data=val_dataset,
                    callbacks=callbacks)
```

Training and evaluating a simple LSTM-based model



TÉCNICO LISBOA

TÉCNICO LISBOA

Deep Learning

SI10 – Deep Learning – Text and Sequences

Reading:

- Ian Goodfellow. ***Deep Learning***. MIT Press, 2016.
- François Chollet. ***Deep Learning With Python***. 2nd Edition, 2017.
- S. Haykin. ***Neural Networks and Learning Machines***. Pearson Education, 2016.

TÉCNICO LISBOA