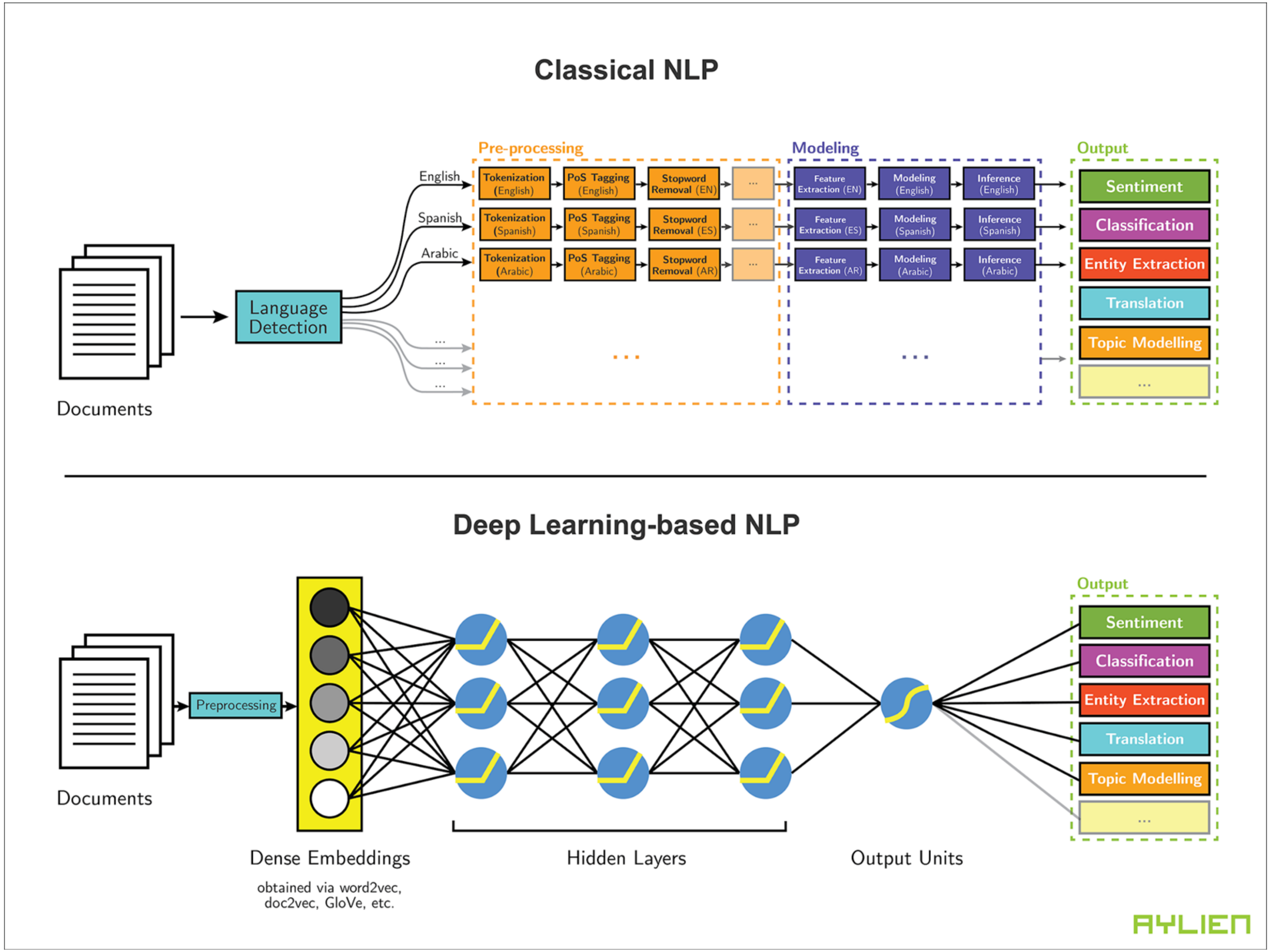
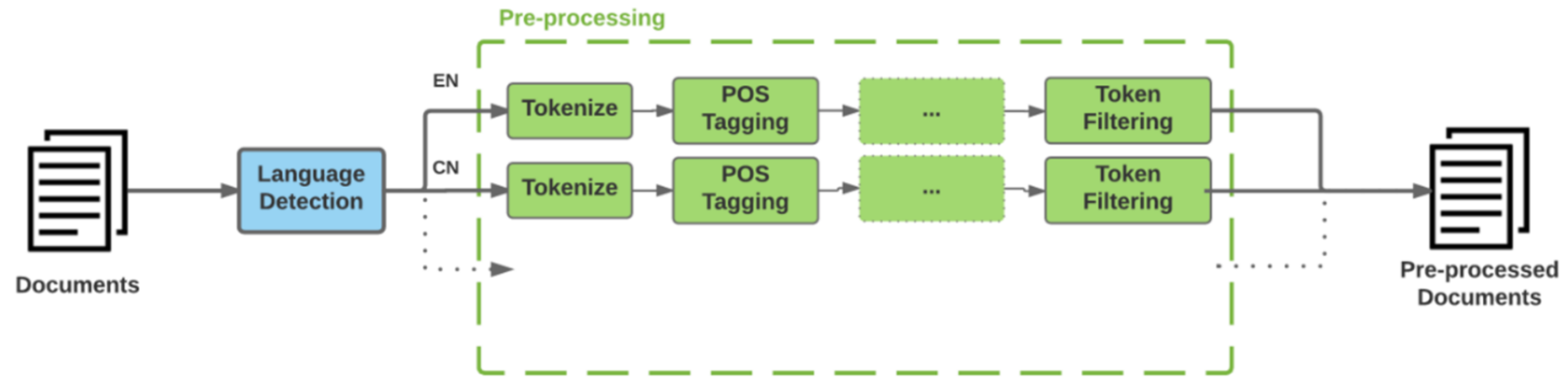
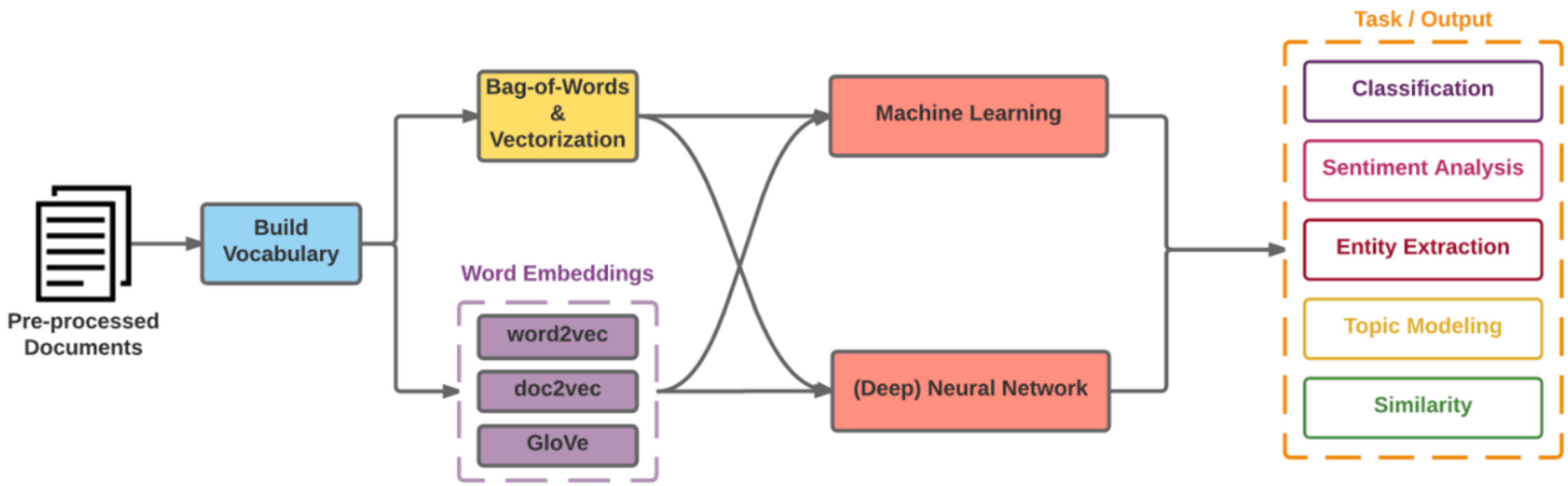
Deep Learning for NLP



Source: http://blog.aylien.com/leveraging-deep-learning-for-multilingual/

Modern NLP Pipeline





Source: https://github.com/fortiema/talks/blob/master/opendata2016sh/pragmatic-nlp-opendata2016sh.pdf

Natural Language Processing (NLP) and Text Mining

- **Raw text**
- **Sentence Segmentation**
- **Tokenization**
- **Part-of-Speech (POS)**
- **Stop word removal**
- **Stemming / Lemmatization**
- **Dependency Parser**
- **String Metrics & Matching**

word’s stem	word’s lemma
am → am	am → be
having → hav	having → have

Source: Nitin Hardeniya (2015), NLTK Essentials, Packt Publishing; Florian Leitner (2015), Text mining - from Bayes rule to dependency parsing

Outline

- Word Embeddings
- Recurrent Neural Networks for NLP
- Sequence-to-Sequence Models
- The Transformer Architecture
- Pretraining and Transfer Learning
- State of the art (SOTA)

Word Embeddings

- Amaçlanan Kelime temsili:
  - Manuel özellik mühendisliği gerektirmeyen
  - ancak anlamsal olarak ("kedi" ve "yavru kedi") ilişkili sözcükler arasında genelleştirmeye izin veren bir temsil.
- Bir sinir ağında kullanmak için bir kelimeyi x girdi vektörüne nasıl kodlamalıyız?
  - one-hot vector: yani, sözlükteki i. sözcüğü, i. giriş konumunda 1 bit ve diğer tüm konumlarda 0 olacak şekilde kodladık. Ancak böyle bir temsil, kelimeler arasındaki benzerliği yakalayamayacaktır.
  - her kelimeyi, kelimenin içinde geçtiği tüm ifadelerin n-gram sayılarının bir vektörü ile temsil edebiliriz. (With a 100,000-word vocabulary, there are 1025 5-grams to keep track of)
  - Bunu daha küçük boyutlu bir vektöre indirgemeliyiz. (bu daha küçük birkaç yüz boyutlu, yoğun vektöre kelime gömme diyoruz)
- **word embedding: a low-dimensional vector representing a word.**
- word embeddings verilerden otomatik olarak öğrenilir.

One-hot encoding

```
'The mouse ran up the clock' =

The      1   [ [0, 1, 0, 0, 0, 0, 0],
mouse    2     [0, 0, 1, 0, 0, 0, 0],
ran      3     [0, 0, 0, 1, 0, 0, 0],
up       4     [0, 0, 0, 0, 1, 0, 0],
the      1     [0, 1, 0, 0, 0, 0, 0],
clock    5     [0, 0, 0, 0, 0, 1, 0] ]

               [0, 1, 2, 3, 4, 5, 6]
```


Source: https://developers.google.com/machine-learning/guides/text-classification/step-3

Word Embeddings

- GloVe (trained on 6 billion words of text)
  100-dimensional word vectors are projected down onto two dimensions

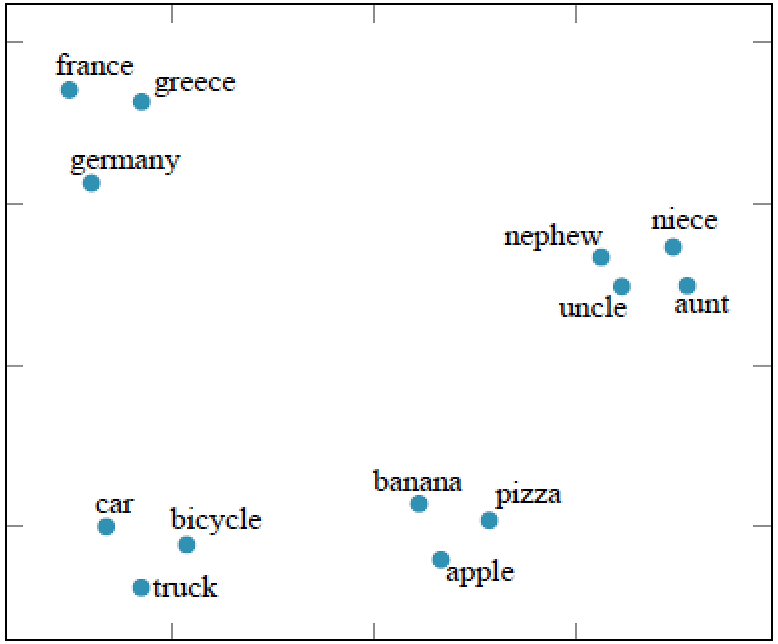


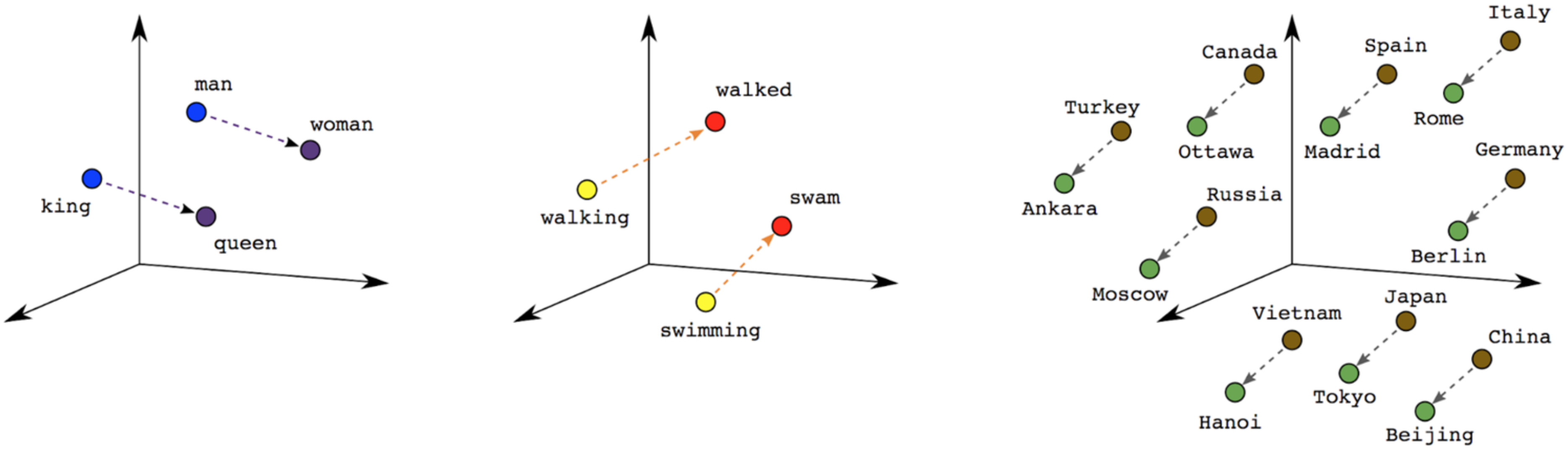
Figure 25.1 Word embedding vectors computed by the GloVe algorithm trained on 6 billion words of text. 100-dimensional word vectors are projected down onto two dimensions in this visualization. Similar words appear near each other.

Word Embedding model answer the question “A is to B as C is to [what]?”

A	B	C	$D = C + (B - A)$	Relationship
Athens	Greece	Oslo	Norway	*Capital*
Astana	Kazakhstan	Harare	Zimbabwe	*Capital*
Angola	kwanza	Iran	rial	*Currency*
copper	Cu	gold	Au	*Atomic Symbol*
Microsoft	Windows	Google	Android	*Operating System*
New York	New York Times	Baltimore	Baltimore Sun	*Newspaper*
Berlusconi	Silvio	Obama	Barack	*First name*
Switzerland	Swiss	Cambodia	Cambodian	*Nationality*
Einstein	scientist	Picasso	painter	*Occupation*
brother	sister	grandson	granddaughter	*Family Relation*
Chicago	Illinois	Stockton	California	*State*
possibly	impossibly	ethical	unethical	*Negative*
mouse	mice	dollar	dollars	*Plural*
easy	easiest	lucky	luckiest	*Superlative*
walking	walked	swimming	swam	*Past tense*

Figure 25.2 A word embedding model can sometimes answer the question “**A** is to **B** as **C** is to [what]?” with vector arithmetic: given the word embedding vectors for the words **A**, **B**, and **C**, compute the vector $\mathbf{D} = \mathbf{C} + (\mathbf{B} - \mathbf{A})$ and look up the word that is closest to **D**. (The answers in column **D** were computed automatically by the model. The descriptions in the “Relationship” column were added by hand.) Adapted from Mikolov *et al.* (2013, 2014).

Word embeddings



Male-Female Verb Tense Country-Capital

Source: https://developers.google.com/machine-learning/guides/text-classification/step-3

application of deep learning to NLP: POS tagging

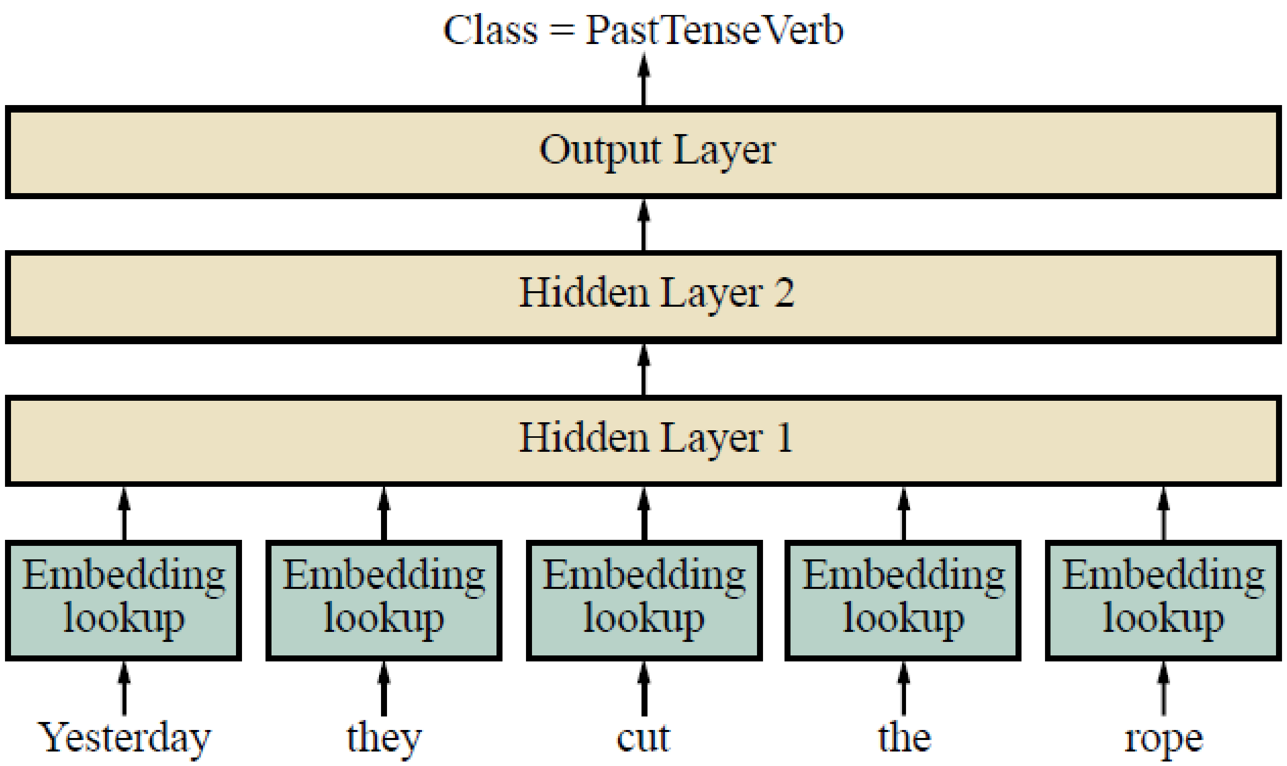


Figure 25.3 Feedforward part-of-speech tagging model. This model takes a 5-word window as input and predicts the tag of the word in the middle—here, *cut*. The model is able to account for word position because each of the 5 input embeddings is multiplied by a different part of the first hidden layer. The parameter values for the word embeddings and for the three layers are all learned simultaneously during training.

Outline

- Word Embeddings
- Recurrent Neural Networks for NLP
- Sequence-to-Sequence Models
- The Transformer Architecture
- Pretraining and Transfer Learning
- State of the art (SOTA)

Recurrent Neural Networks for NLP

- Tek tek kelimeler için iyi bir temsil yöntemi var ancak dil, sözcüklerin bağlamının önemli olduğu bir kelime dizisinden oluşur.
- Her seferinde bir veri olmak üzere zaman serisi verilerini işlemek için tasarlanmış tekrarlayan sinir ağı (RNN), her seferinde bir kelime olmak üzere dili işlemek için de kullanılabilir.

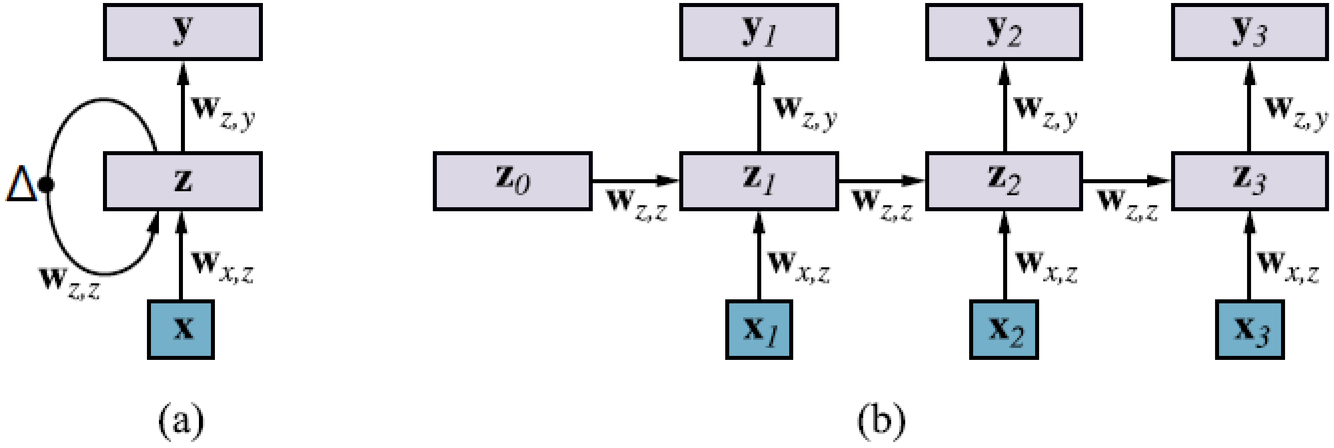
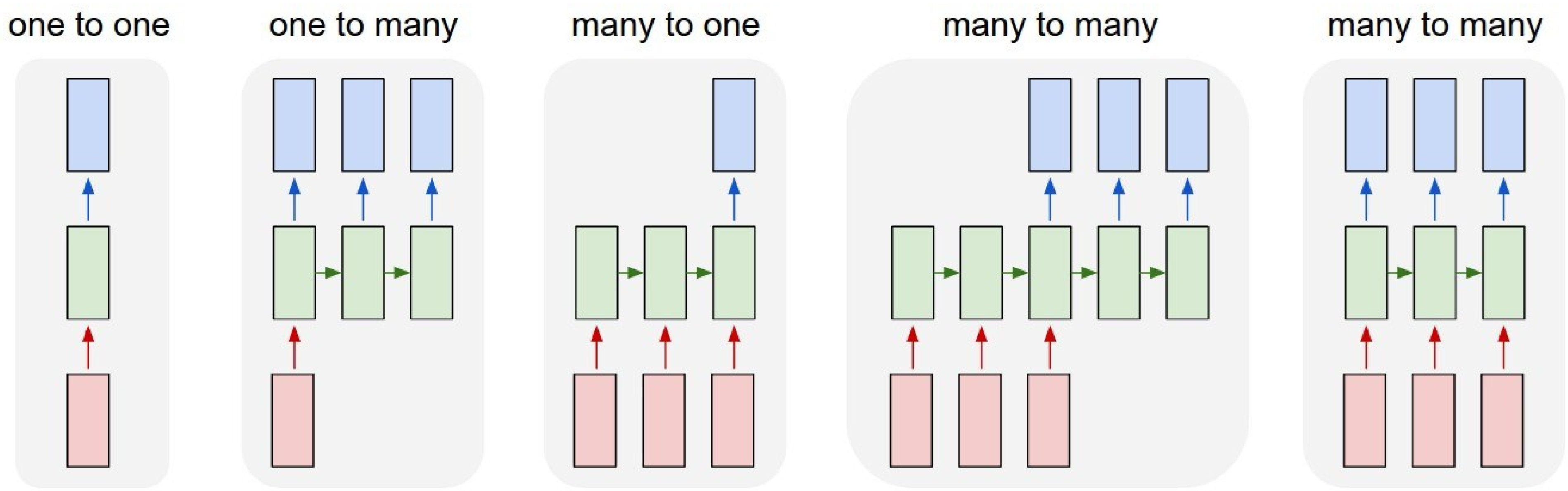


Figure 25.4 (a) Schematic diagram of an RNN where the hidden layer **z** has recurrent connections; the Δ symbol indicates a delay. Each input **x** is the word embedding vector of the next word in the sentence. Each output **y** is the output for that time step. (b) The same network unrolled over three timesteps to create a feedforward network. Note that the weights are shared across all timesteps.

LSTMs for NLP tasks

- LSTM, bir zaman adımından diğerine mesajı kusurlu bir şekilde yeniden üretme problemi olmayan geçit ünitelerine(gating units) sahip bir tür RNN'dir.
- LSTM girdinin bazı kısımlarını hatırlamayı, onu bir sonraki zaman adımına kopyalamayı ve diğer kısımlarını unutmayı seçebilir.
- Aşağıdaki metni işlemek üzere bir dil modeli düşünün:
- «The athletes, who all won their local qualifiers and advanced to the finals in Tokyo, now ...»
- Modele bir sonraki kelimenin hangisinin daha olası olduğunu sorsak
  - “compete” veya “competes” diye sorsak, “The athletes” konusuna uyduğu için “complete” seçmesini beklerdik.
- Bir LSTM, söz konusu kişi için gizli bir özellik oluşturmayı öğrenebilir ve böyle bir seçim yapmak için gerekli olana kadar bu özelliği değiştirmeden kopyalayabilir.
- Normal bir RNN (veya bu konuda bir n-gram modeli), özne ve fiil arasında birçok araya giren kelimelerden kurulu uzun cümlelerde sıklıkla karışır.

LSTM Recurrent Neural Network



Traditional Neural Network

Music Generation

Sentiment Classification

Name Entity Recognition

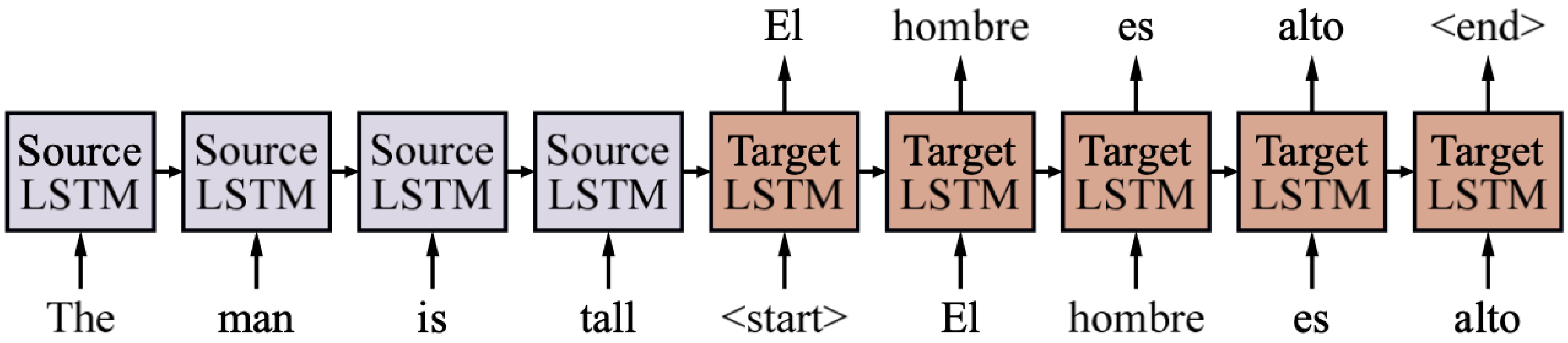
Machine Translatio n

Source: https://github.com/Vict0rSch/deep_learning/tree/master/keras/recurrent

Outline

- Word Embeddings
- Recurrent Neural Networks for NLP
- Sequence-to-Sequence Models
- The Transformer Architecture
- Pretraining and Transfer Learning
- State of the art (SOTA)

Sequence-to-Sequence model



Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

Attention

Standard target RNN:

$h_i = RNN(h_{i-1};x_i)$;

Target RNN for attentional sequence-to-sequence models

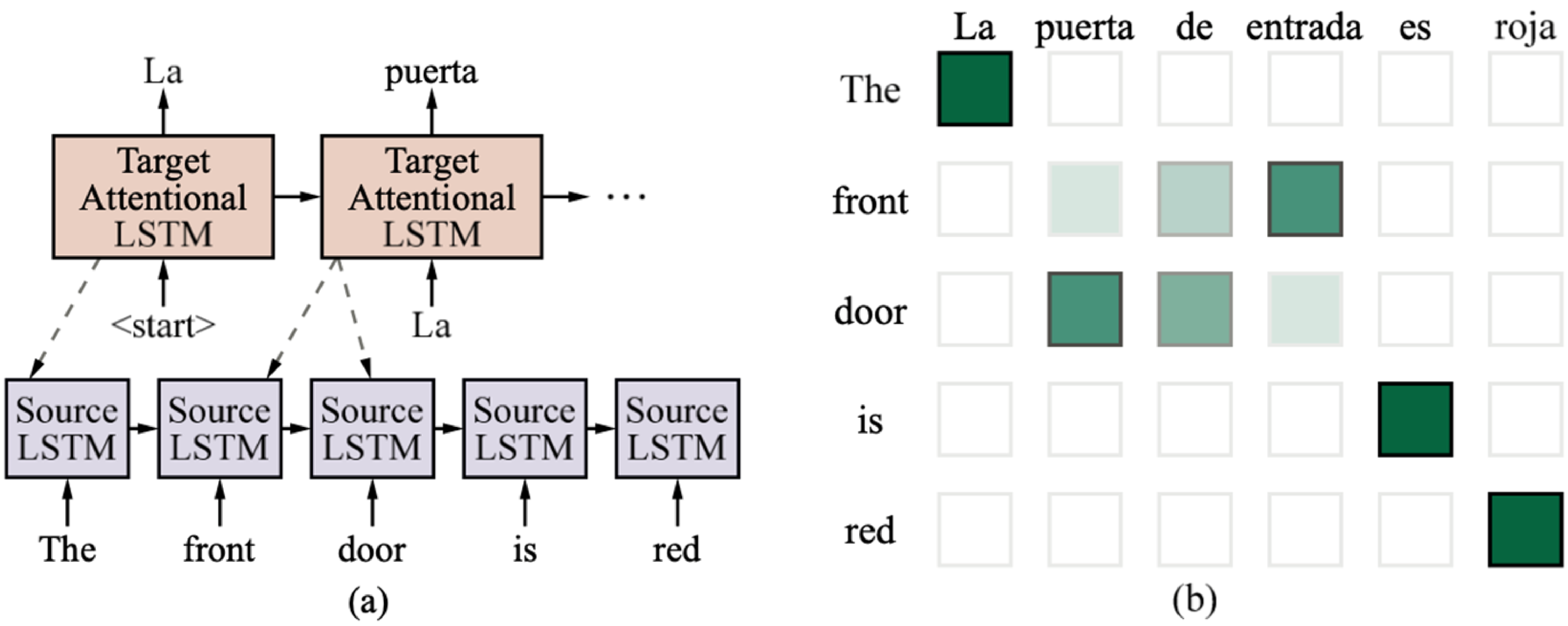
$h_i = RNN(h_{i-1}; [x_i; c_i])$

Burada, $[x_i; c_i]$ girdi ve bağlam vektörlerinin birleşimidir.

$$
\begin{aligned}
r_{ij} &= \mathbf{h}_{i-1} \cdot \mathbf{s}_j \\
a_{ij} &= e^{r_{ij}} / (\sum_k e^{r_{ik}}) \\
\mathbf{c}_i &= \sum_j a_{ij} \cdot \mathbf{s}_j
\end{aligned}
$$

- h_{i-1} i zaman adımında kelimeyi tahmin etmek için kullanılacak olan hedef RNN vektörüdür,
- s_j kelime j (veya zaman adımı j) için kaynak RNN vektörünün çıktısıdır.
  - Hem h_{i-1} hem de s_j d boyutlu vektörlerdir, (d:hidden size)
- r_{ij} mevcut hedef durum ile kaynak kelime j arasındaki ham "dikkat puanı"dır.
  - Bu puanlar daha sonra tüm kaynak sözcükler üzerinde bir softmax kullanılarak bir olasılık değeri olarak normalleştirilir.
- Son olarak, bu olasılıklar, kaynak RNN vektörlerinin, ci (başka bir d-boyutlu vektör) ağırlıklı ortalamasını oluşturmak için kullanılır.
  - Dikkat bileşeninin kendisinin öğrenilmiş ağırlıkları yoktur.
  - Programcı, hangi bilgilerin ne zaman kullanılacağını dikte etmez; model neyi kullanacağını öğrenir.

Attentional Sequence-to-Sequence model for English-to-Spanish translation



Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

Outline

- Word Embeddings
- Recurrent Neural Networks for NLP
- Sequence-to-Sequence Models
- The Transformer Architecture
- Pretraining and Transfer Learning

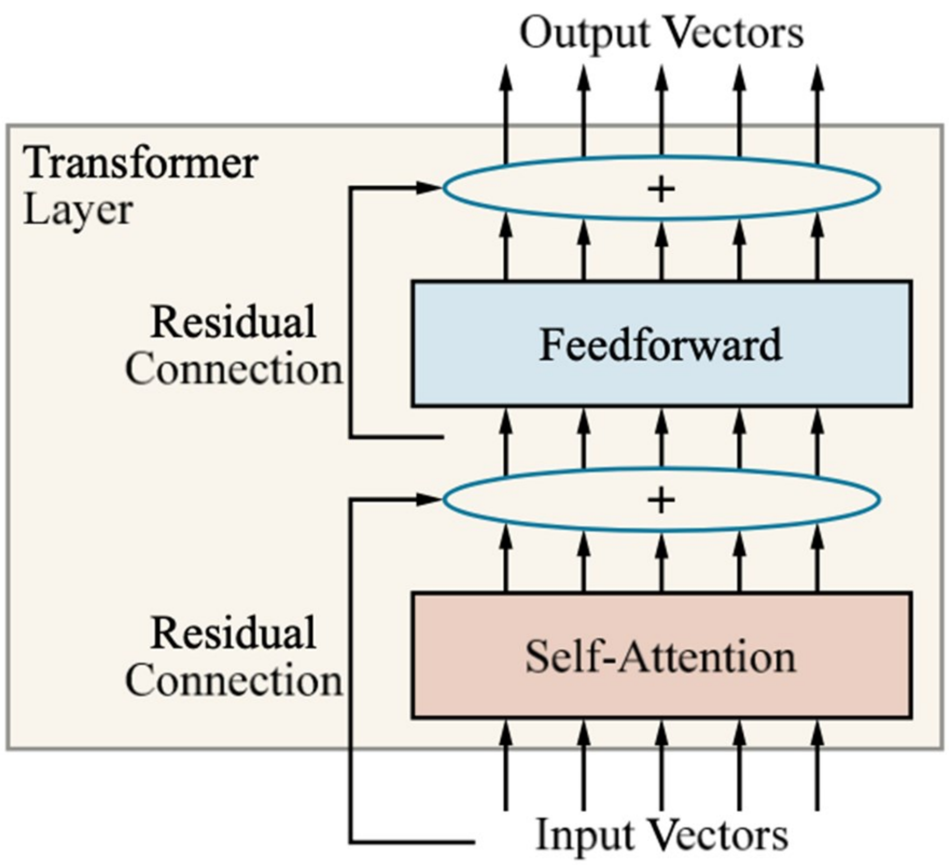
The Transformer Architecture

The influential article “Attention is all you need” (Vaswani et al., 2018) introduced the transformer architecture, which uses a self-attention mechanism that can model long-distance context without a sequential dependency

https://jalammar.github.io/illustrated-transformer/

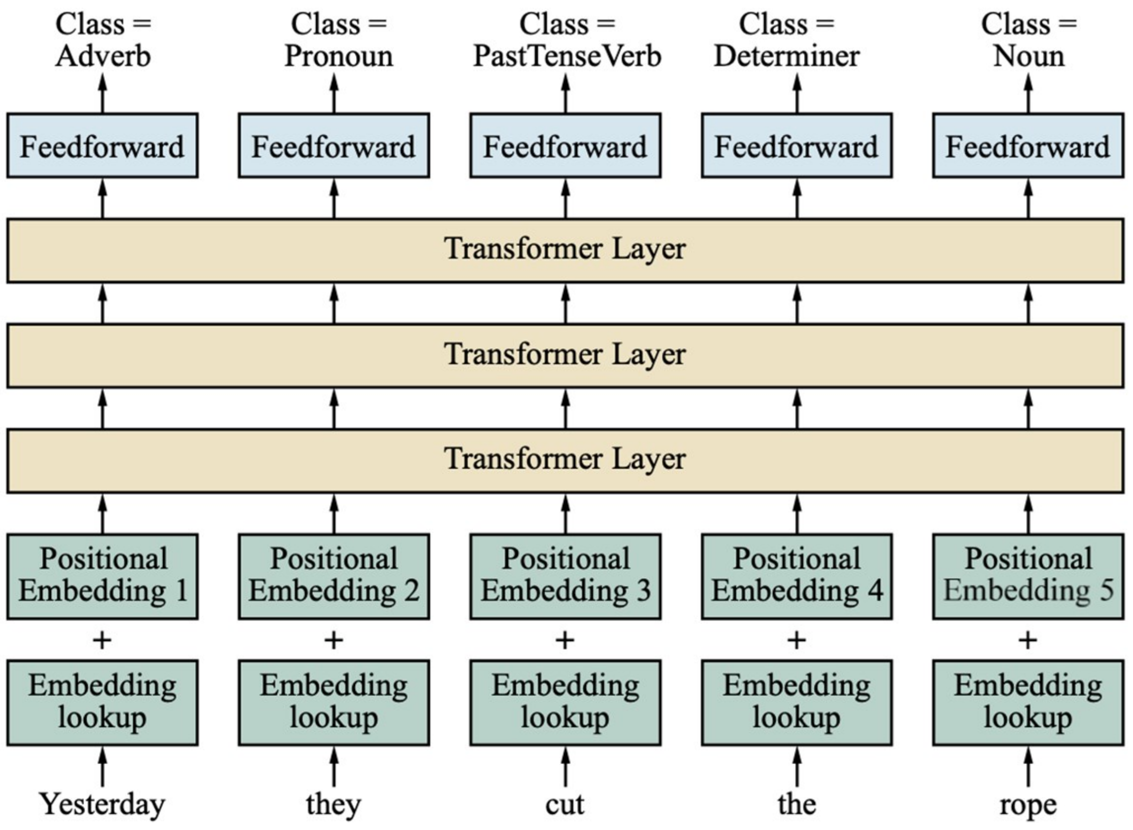
Single-layer Transformer

consists of self-attention, a feedforward network, and residual connection



Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

Transformer Architecture for POS Tagging

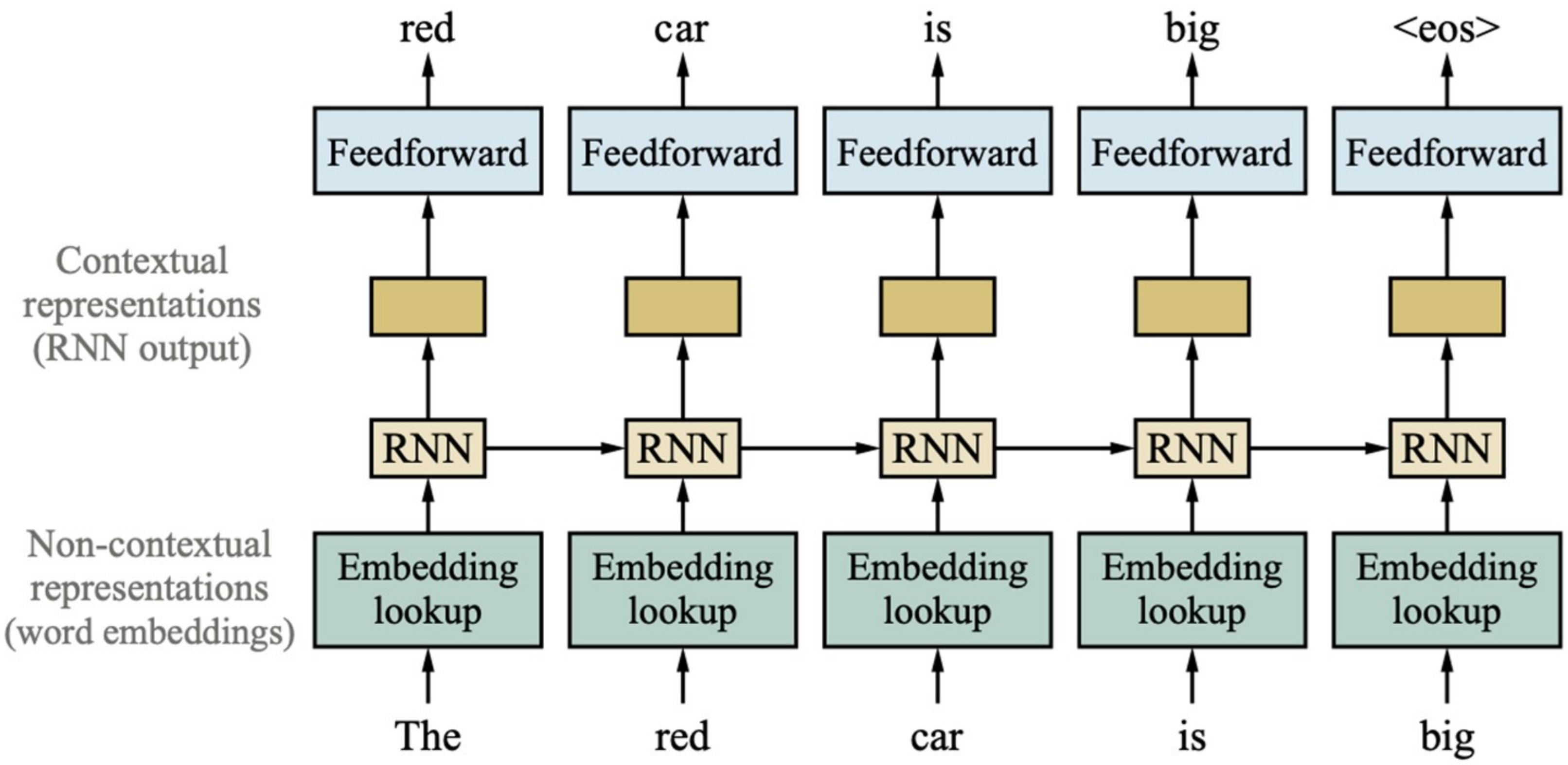


Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

Outline

- Word Embeddings
- Recurrent Neural Networks for NLP
- Sequence-to-Sequence Models
- The Transformer Architecture
- Pretraining and Transfer Learning

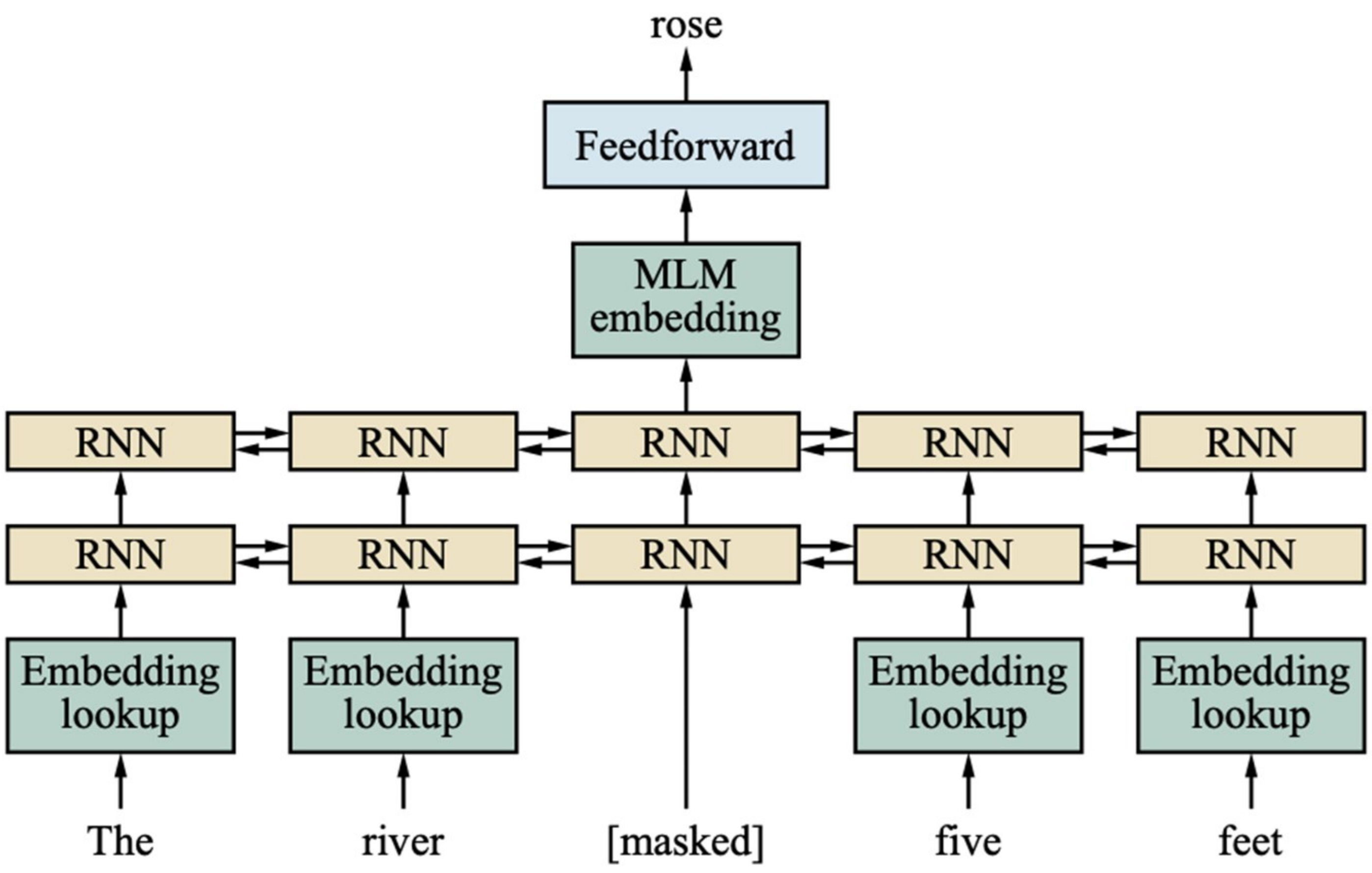
Training Contextual Representations
using a left-to-right Language Model



Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

Masked Language Modeling:

Pretrain a Bidirectional Model



Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

References

- Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson.
- Aurélien Géron (2019), Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 2nd Edition, O'Reilly Media.
- Dipanjan Sarkar (2019), Text Analytics with Python: A Practitioner's Guide to Natural Language Processing, Second Edition. APress. https://github.com/Apress/text-analytics-w-python-2e
- Benjamin Bengfort, Rebecca Bilbro, and Tony Ojeda (2018),
Applied Text Analysis with Python, O'Reilly Media.
https://www.oreilly.com/library/view/applied-text-analysis/9781491963036/
- Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Céspedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, Ray Kurzweil (2018). Universal Sentence Encoder. arXiv:1803.11175.
- Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego , Steve Yuan, Chris Tar, Yun-hsuan Sung, Ray Kurzweil (2019). Multilingual Universal Sentence Encoder for Semantic Retrieval.
- Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang (2020). "Pre-trained Models for Natural Language Processing: A Survey." arXiv preprint arXiv:2003.08271.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805.
- Jay Alammar (2019), The Illustrated Transformer, http://jalammar.github.io/illustrated-transformer/
- Jay Alammar (2019), A Visual Guide to Using BERT for the First Time,
http://jalammar.github.io/a-visual-guide-to-using-bert-for-the-first-time/
- Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- HuggingFace (2020), Transformers Notebook, https://huggingface.co/transformers/notebooks.html
- The Super Duper NLP Repo, https://notebooks.quantumstat.com/
- Min-Yuh Day (2021), Python 101, https://tinyurl.com/aintpupython101