RNN

Sieci rekurencyjne i modelowanie sekwencji

Modelowanie sekwencji

Sieć jednokierunkowa realizuje odwzorowanie jeden do jednego

$$F(\mathbf{x}_i; \theta) = \mathbf{y}_i$$

Przy prezentacji sekwencji $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t$ na wejście MLP każda ramka $\mathbf{x}_i$ jest niezależna od innych ramek, więc sieć nie może modelować zależności czasowych.

$$\begin{aligned} F(\mathbf{x}_0; \theta) &= \mathbf{y}_0 \\ F(\mathbf{x}_1; \theta) &= \mathbf{y}_1 \\ &\ldots \\ F(\mathbf{x}_t; \theta) &= \mathbf{y}_t \end{aligned}$$

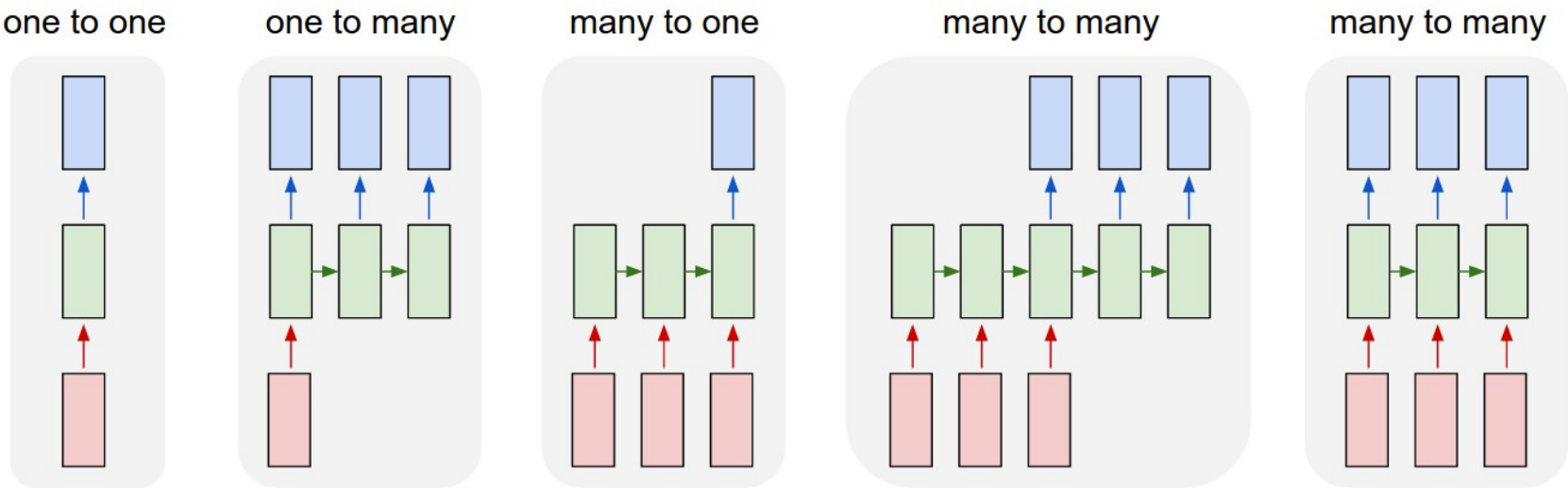
Modelując szeregi czasowe chcemy uzyskać odpowiedź na podstawie sekwencji historycznych wartości

$$F(\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t; \theta) = \mathbf{y}$$

Problemy modelowania sekwencji

- Dane sekwencyjne lub zależne od czasu: sygnał audio, tekst, ruch obiektów, EEG, ...
- Modelowanie sekwencji: przewidywanie kolejnych wartości szeregu (np. przewidywanie pogody, notowań giełdowych), generowanie tekstu (chatboty), generowanie muzyki, sterowanie ruchem robota, ...
- Przetwarzanie sekwencji: maszynowe tłumaczenie, rozpoznawanie mowy, rozpoznawanie pisma, przetwarzanie strumieni wideo, ...
- Klasyfikacja sygnałów czasowych, etykietowanie obrazów, wykrywanie wzorców w sekwencjach

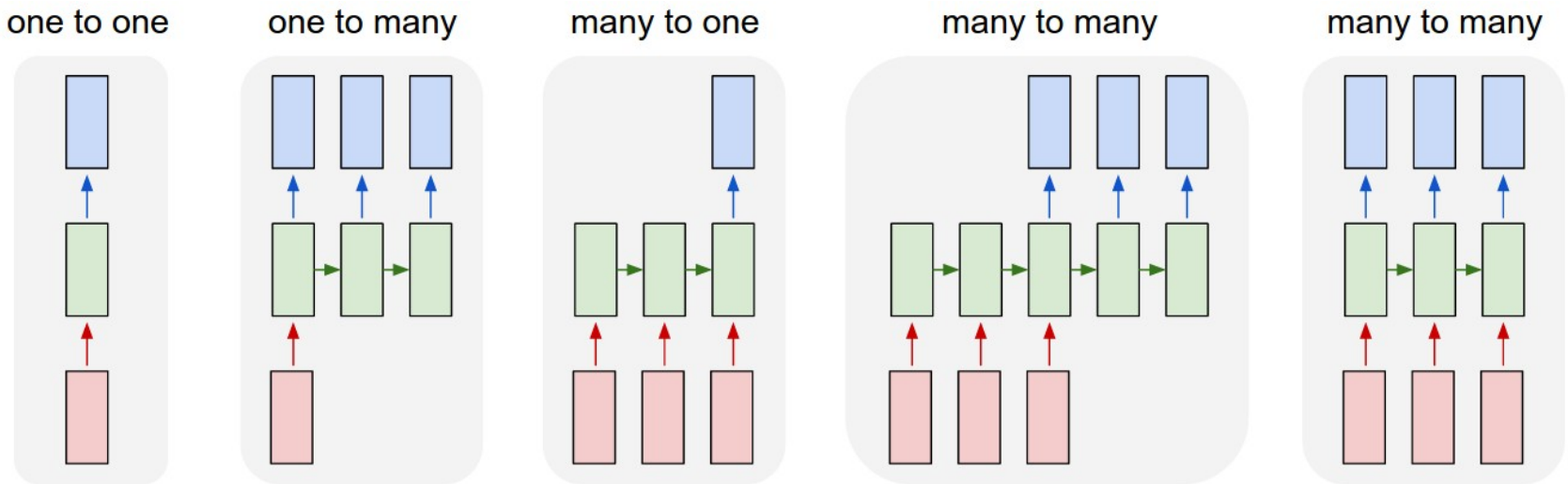
Modelowanie sekwencji



- *one-to-one*: pojedynczy sygnał wejściowy produkuje pojedynczy sygnał wyjściowy, brak modelowania relacji zależnych od czasu, np. klasyfikacja obrazów
- *one-to-many*: pojedyncze wejście generuje sekwencję na wyjściu, np. tworzenie opisów obrazów, generowanie muzyki, generowanie tekstu
- *many-to-one*: sekwencja wejściowa produkuje pojedynczy sygnał wyjściowy, np. klasyfikacja dokumentów, analiza sentymentu

Rys: ☞ Andrej Karpathy, Hacker's guide to Neural Networks

Modelowanie sekwencji



many-to-many: wejście i wyjście sekwencyjne

- bez dopasowania ramek, długość sekwencji wejściowej jest różna od długości sekwencji wyjściowej, np. tłumaczenie maszynowe, transkrypcja mowy
- z dopasowaniem ramek (*alignment*), np: etykietowanie ramek wideo, klasyfikacja fonemów w sygnale mowy (model akustyczny)

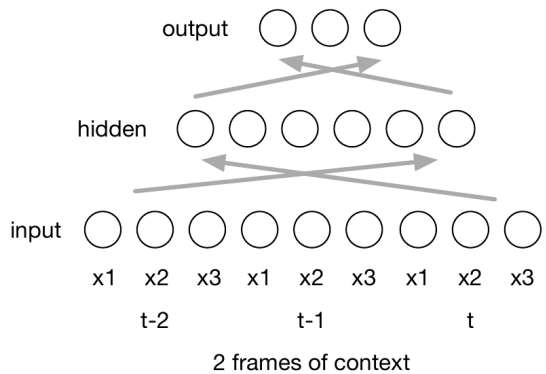
Rys: ☞ Andrej Karpathy, Hacker's guide to Neural Networks

Modelowanie sekwencji sieciami jednokierunkowymi

Naiwne podejście stosowane w MLP: dodanie **stałego w czasie kontekstu** do aktualnego wejścia $\mathbf{x}_i$

Przykład: po 2 ramki kontekstu z lewej i prawej (poprzednie i przyszłe wektory wejściowe)

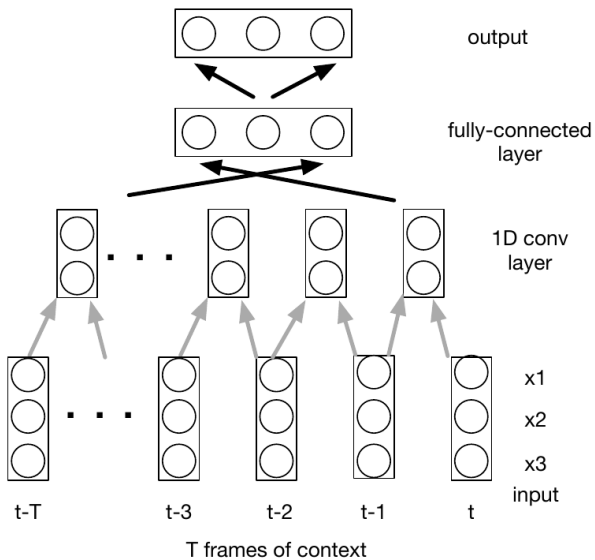
$$\hat{\mathbf{x}}^t \leftarrow [\mathbf{x}^{t-2}, \mathbf{x}^{t-1}, \mathbf{x}^t, \mathbf{x}^{t+1}, \mathbf{x}^{t+2}]$$

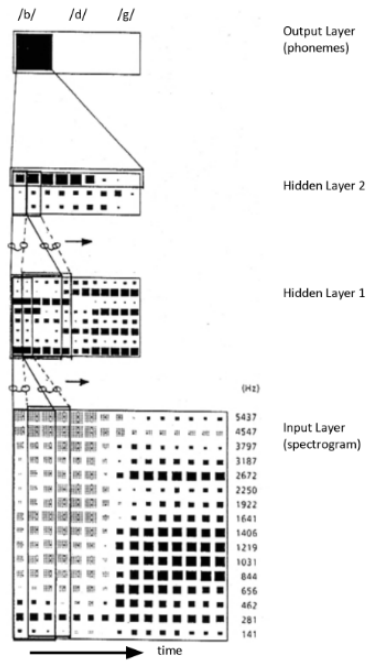


- zwiększenie wymiaru wejściowego istotnie zwiększa złożoność modelu, więc potrzebna jest większa liczba próbek uczących
- modelowanie tylko zależności krótkiego zasięgu (*short-term memory*), ograniczonych wielkością okna kontekstu

Uczenie sekwencji z MLP i CNN

Sieci CNN: splot względem wymiaru czasu (splot 1D), stały kontekst odpowiada wielkości efektywnego pola recepcyjnego w wymiarze czasu.



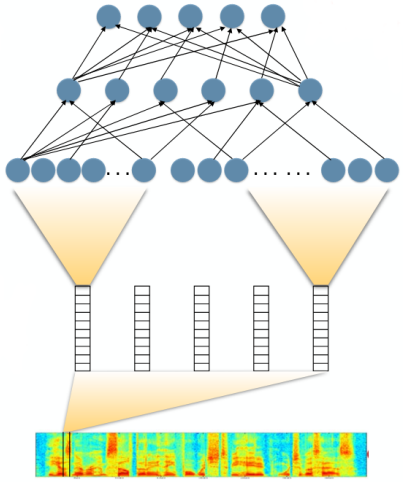


TDNN (Waibel, 1987), *time-delay neural network*, model ze splotem 1D w domenie czasu.

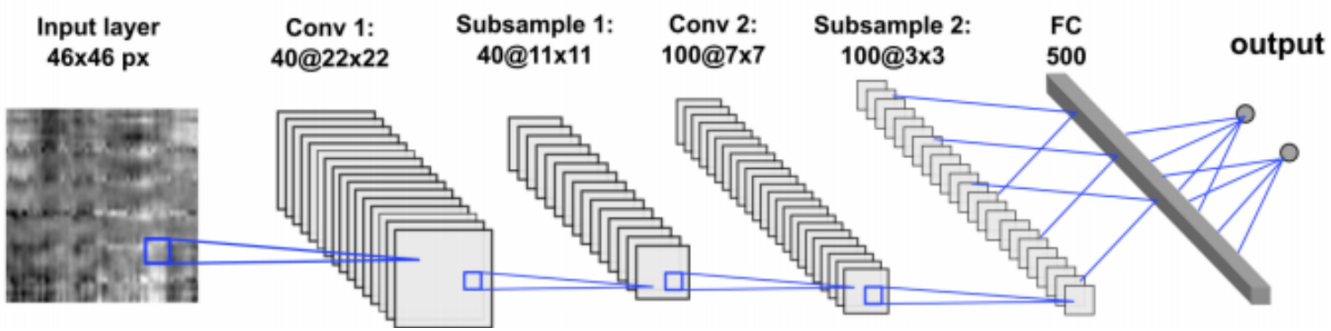
Źródło grafik: Steve Renals, Machine Learning Practical — MLP Lecture 9
Alexander Waibel, Phoneme Recognition Using Time-Delay Neural Networks, 1987.

Przykłady

Model akustyczny: rozpoznawanie fonemów z nagrań audio



Analiza sygnałów EEG: sygnał w postaci obrazów (elektroda $\times$ czas) lub widma mocy, klasyfikacja obrazów siecią CNN



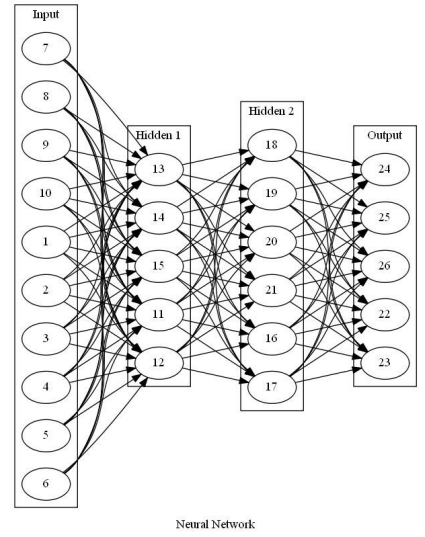
Nurse, E., Mashford, B. S., Yepes, A. J., Kiral-Kornek, I., Harrer, S., & Freestone, D. R. (2016). Decoding EEG and LFP Signals using Deep Learning: Heading TrueNorth

RNN - Sieci rekurencyjne

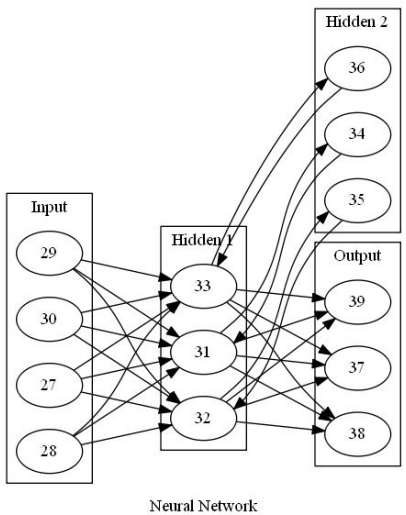
- **Sieci jednokierunkowe** używają jedynie kontekstu o skończonej długości, nie są w stanie wykryć zależności w dłuższych okresach czasu (*long-term*)

- **Sieci rekurencyjne (RNN)** posiadają połączenia do wcześniejszych warstw, wyjście w chwili t zależy od stanu z czasu $t-1$ oraz z kroków poprzednich dzięki rekurencji

Sieć jednokierunkowa

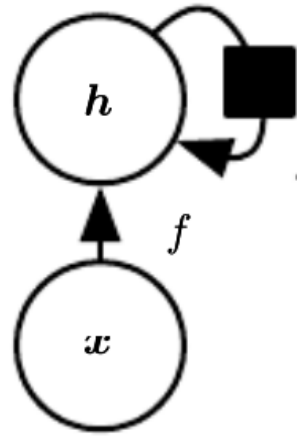


Sieć rekurencyjna Elmana



Stan ukryty jednostek RNN

$$\mathbf{h}^{(t)} = f(\mathbf{h}^{(t-1)}, \mathbf{x}^{(t)}; \boldsymbol{\theta})$$

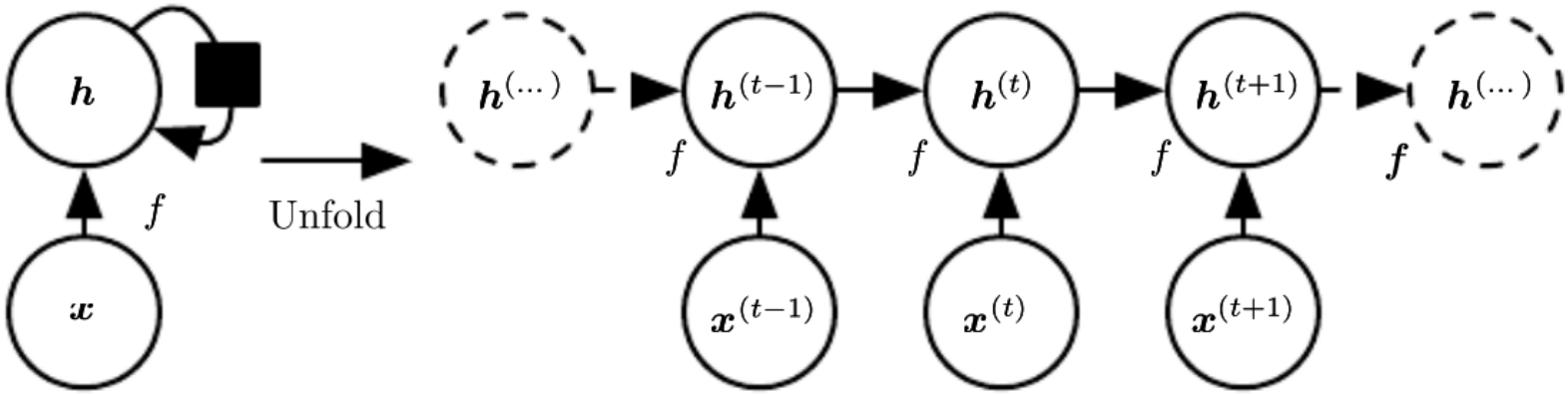


- neurony z połączeniami rekurencyjnymi działają jak pamięć, potencjalnie są w stanie modelować relacje występujące z dowolnie długim odstępem czasowym (nieskończony kontekst)

$$\mathbf{h}^{(t)} = g(\mathbf{x}^{(t)}, \mathbf{x}^{(t-1)}, \ldots, \mathbf{x}^{(1)})$$

- wektor $\mathbf{h}^t$ (aktywacje neuronów warstwy rekurencyjnej) tworzy skompresowaną reprezentację sekwencji
- stan początkowy $\mathbf{h}^0$ może być interpretowany jako dodatkowy sygnał wejściowy, typowo inicjowany wartością zero

Rozwinięcie rekurencji w czasie

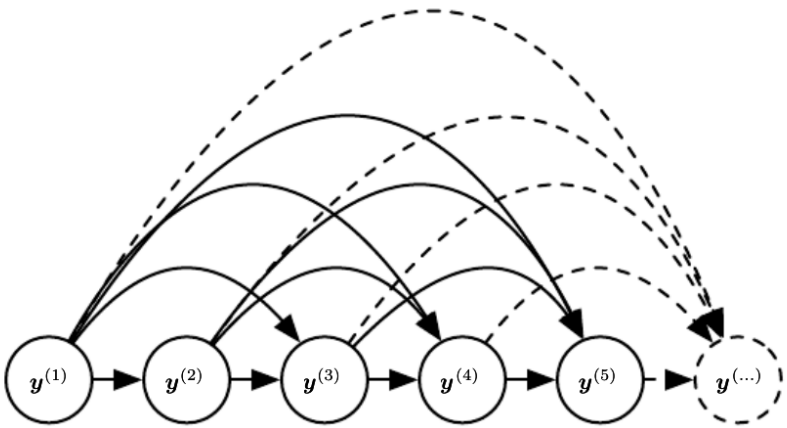


Rozwinięcie sieci RNN do postaci sieci jednokierunkowej, gdzie wagi połączeń są współdzielone.

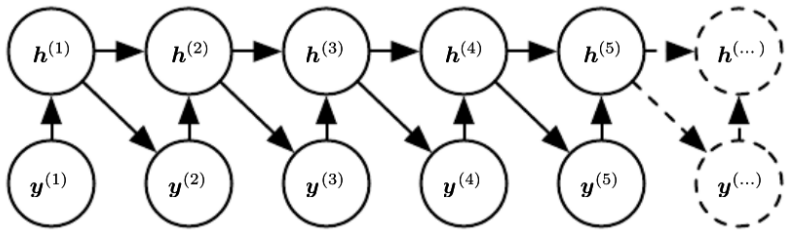
Głębokie uczenie w sieciach RNN: od pierwszego sygnału $\mathbf{x}^1$ do wyjścia $\mathbf{h}^t$ jest wiele warstw

Efektywność kodowania

- Sieć RNN z rekurencją w warstwach ukrytych spełnia wymagania **uniwersalnej maszyny Turinga**
- Ilość parametrów opisujących stan **h** ma wpływ na wielkość pamięci, ale nie musi zależeć od długości sekwencji

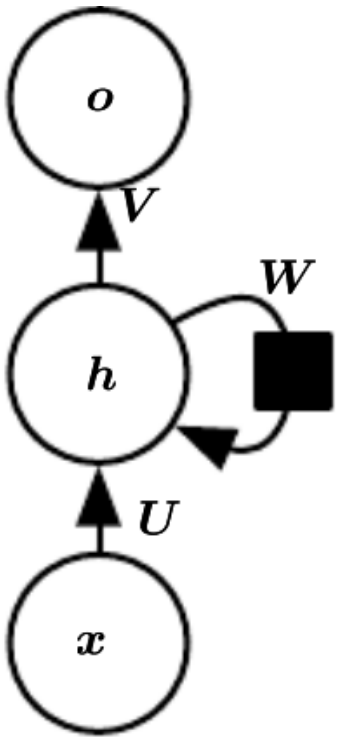


pełny model sekwencji, ilość parametrów $O(k^{\tau})$



RNN, stan obecny zależy pośrednio od stanów z przeszłości, ilość parametrów stała względem długości sekwencji $O(1)$

Sieć z jedną warstwą rekurencyjną



$$o_i^{(t)} = softmax_i \left(\sum_{r=1}^{k} v_{ir} h_r^{(t)} + b_i \right)$$

$$h_i^{(t)} = \sigma \left(\sum_{j=1}^{d} u_{ij} x_j^{(t)} + \sum_{r=1}^{k} w_{ir} h_r^{(t-1)} + c_i \right)$$

To samo zwięźlej w zapisie macierzowym

$$\mathbf{o}^{(t)} = softmax \left(\mathbf{V}\mathbf{h}^{(t)} + \mathbf{b} \right)$$

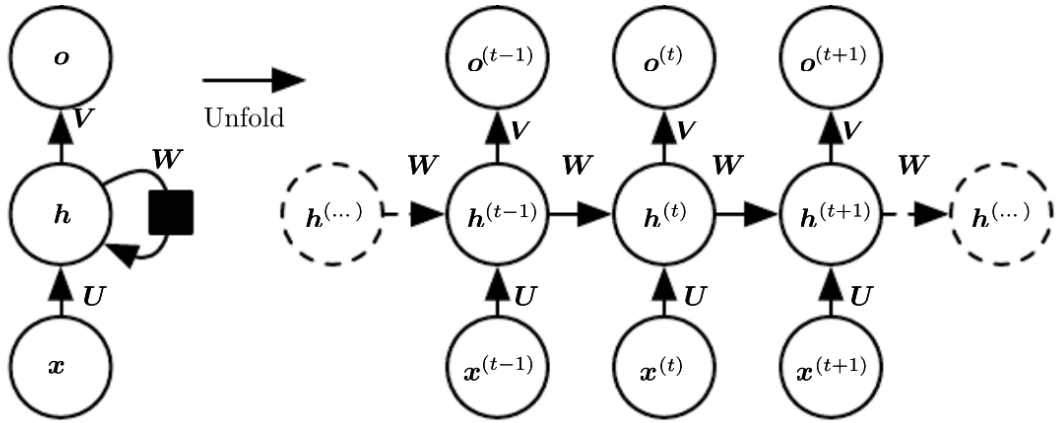
$$\mathbf{h}^{(t)} = \sigma \left(\mathbf{U}\mathbf{x}^{(t)} + \mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{c} \right)$$

$\mathbf{x}^{(t)} \in \mathbb{R}^d$ sygnał wejściowy w chwili t
$\mathbf{U} \in \mathbb{R}^{d \times k}$, $\mathbf{W} \in \mathbb{R}^{k \times k}$
d ilość wejść sieci
k ilość neuronów w warstwie ukrytej

Źródło grafiki: Goodfellow, Deep Learning, 2016

Wsteczna propagacja w czasie

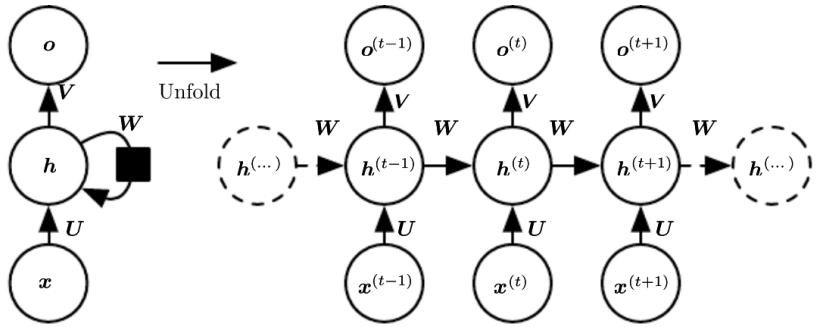
Rozwinięta sieć odpowiada głębokiej sieci jednokierunkowej ze współdzielonymi wagami $\mathbf{W}, \mathbf{V}, \mathbf{U}$



BPTT *backpropagation through time* - trening za pomocą wstecznej propagacji sieci „rozwiniętej w czasie", odbywa się analogicznie jak w jednokierunkowych sieciach.
Aktualizacja wagi w chwili t wymaga:

- całej historii sygnałów przeszłych $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(t)}$
- historii stanów neuronów $\mathbf{h}^{(1)}, \ldots, \mathbf{h}^{(t)}$ i ich pochodnych $\mathbf{h'}^{(1)}, \ldots, \mathbf{h'}^{(t)}$

Wsteczna propagacja w czasie



Funkcja kosztu dla całej sekwencji (model *many-to-many*):

$$L\left(\{\mathbf{x}^{(1)},\mathbf{x}^{(2)},\ldots,\mathbf{x}^{(\tau)}\},\{\mathbf{y}^{(1)},\mathbf{y}^{(2)},\ldots,\mathbf{y}^{(\tau)}\}\right)=\frac{1}{\tau}\sum_{t=1}^{\tau}L^{(t)}$$

$L^{(t)}$ - koszt w kroku t (np. cross-entropy) zależny od $\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(t)}$

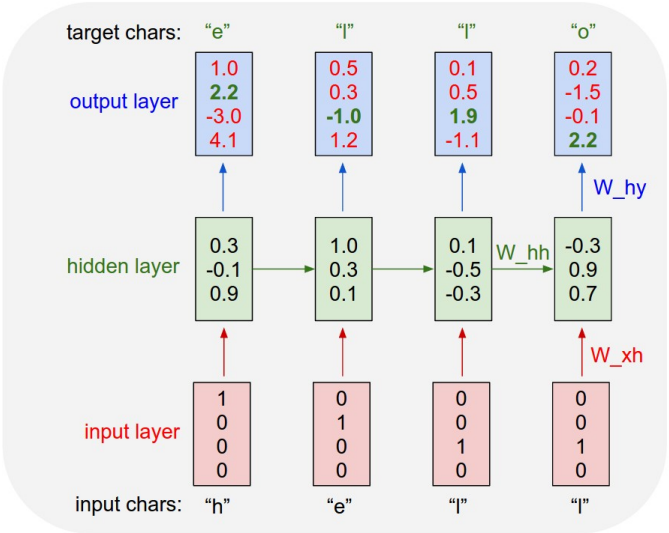
$$\partial_w L=\sum_{t=1}^{\tau}\partial_w L\left(y^{(t)},o^{(t)}\right)$$

Aktualizacje wag są akumulowane dla sygnału błędu propagowanego w kolejnych krokach czasu. Gradient w BPTT zawiera człony łańcuchowe przez stany ukryte i współdzielenie wag w czasie.

Źródło grafiki: Goodfellow, Deep Learning, 2016

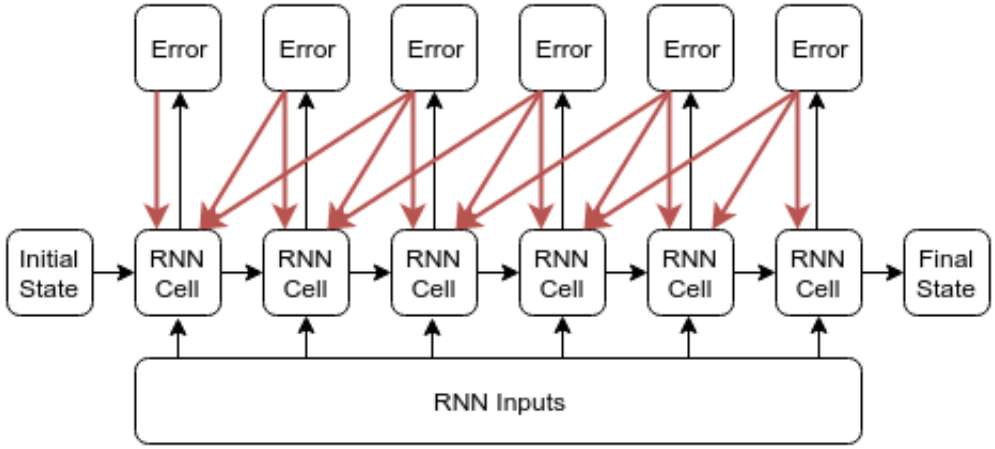
Przykład: CharRNN

- model języka oparty na znakach, wejście i wyjście stanowi sekwencja znaków
- znaki kodowane z pomocą wektorów binarnych o długości równej liczbie wszystkich znaków w alfabecie (kodowanie *one-hot*)
- problem klasyfikacji przewidywania kolejnego znaku w sekwencji $f\left(x^{(t)}\right) \longrightarrow x^{(t+1)}$
- wyjście softmax modeluje rozkład prawdopodobieństwa $P\left(x^{(t+1)} | x^{(1)}, x^{(2)}, \ldots, x^{(t)}\right)$



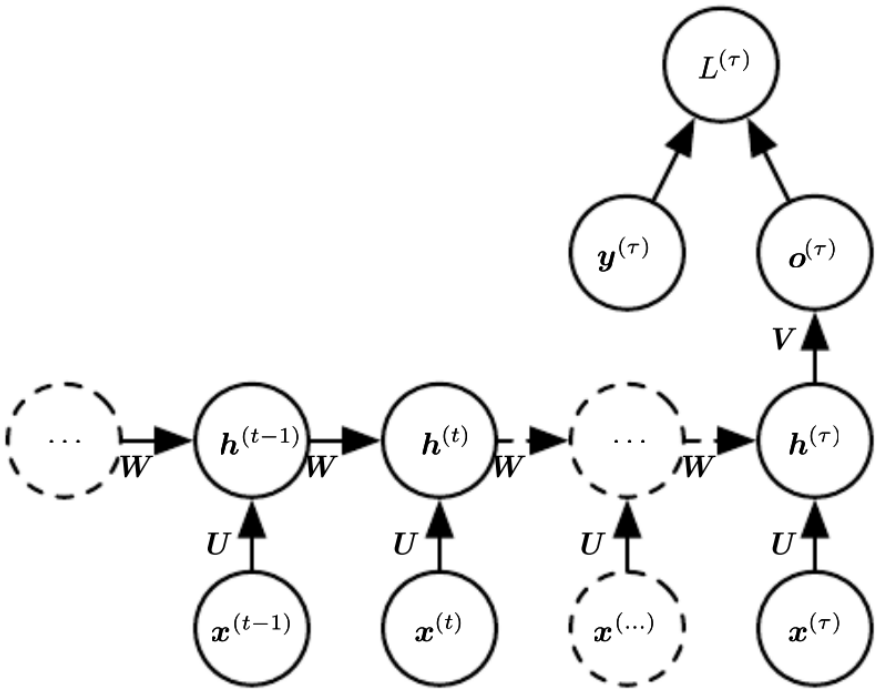
☞ The Unreasonable Effectiveness of Recurrent Neural Networks

Wsteczna propagacja w czasie



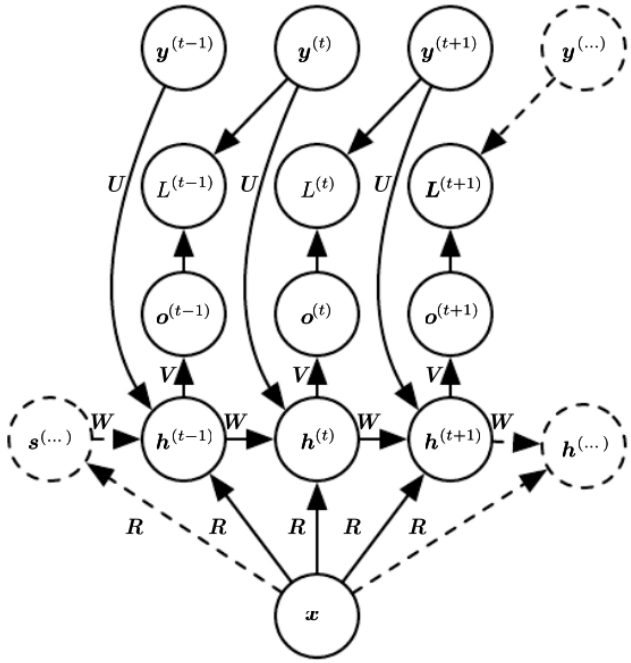
- Rozwinięta w czasie sieć tworzy głęboką sieć neuronową ze wszystkimi tego konsekwencjami (zanikający gradient dla relacji odległych w czasie, wybuchający gradient)
- Obliczenia dla długich sekwencji są kosztowne
- Nie zawsze potrzebujemy informacji z odległej przeszłości
- **Truncated BPTT** - ograniczenie długości sekwencji wstecznej propagacji do stałej wartości T (np. $T = 20$). Gradient dla zależności z dalszej przeszłości $t > T$ jest obcinany do 0
- problem z wykrywaniem odległych relacji (*long-term*) przez obcięcie T oraz zanikający gradient

Pojedyncze wyjście podsumowujące sekwencję



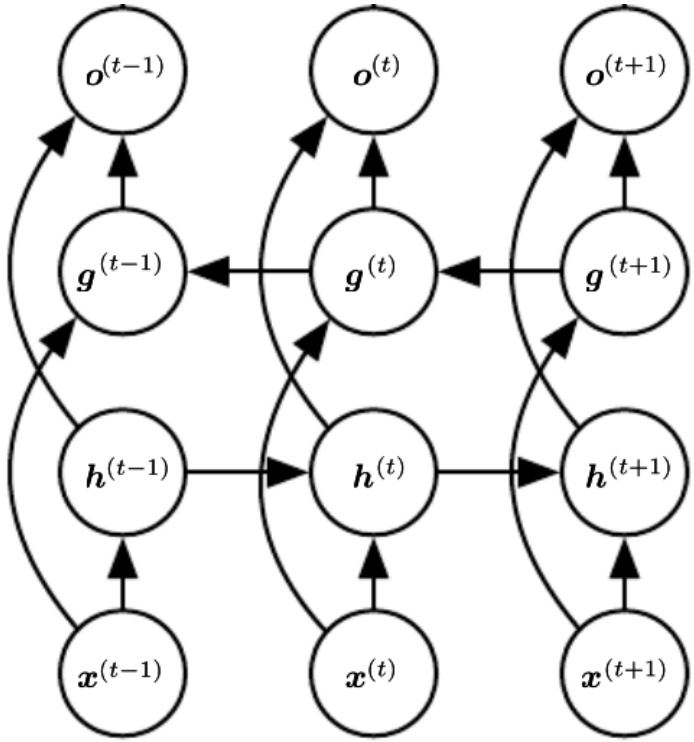
Sieć produkuje wyjście po obejrzeniu całej sekwencji,
np. klasyfikacja wpisów w portalach społecznościowych
(tzw. analiza sentymentów)

Pojedyncze wejście generujące sekwencję



- Sieć produkuje sekwencję wyjść przy prezentacji stałego wzorca (np. etykietowanie zdjęć, opis sceny)
- Sygnał wejściowy **x** pełni rolę kontekstu
- Modelowanie sekwencji $\mathbf{y}^{(t)} = f(\mathbf{y}^{(t-1)}; \mathbf{x})$

Dwukierunkowa sieć RNN



Sieci dwukierunkowe RNN łączą warstwy z rekurencją zależną od sygnału przeszłego **h** z rekurencją zależną od przyszłego sygnału **g**

Schuster, Mike, and Kuldip K. Paliwal. "Bidirectional recurrent neural networks.Śignal Processing, 1997

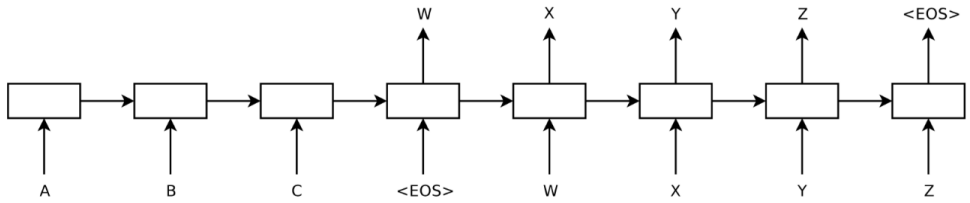
Dwukierunkowa sieć RNN

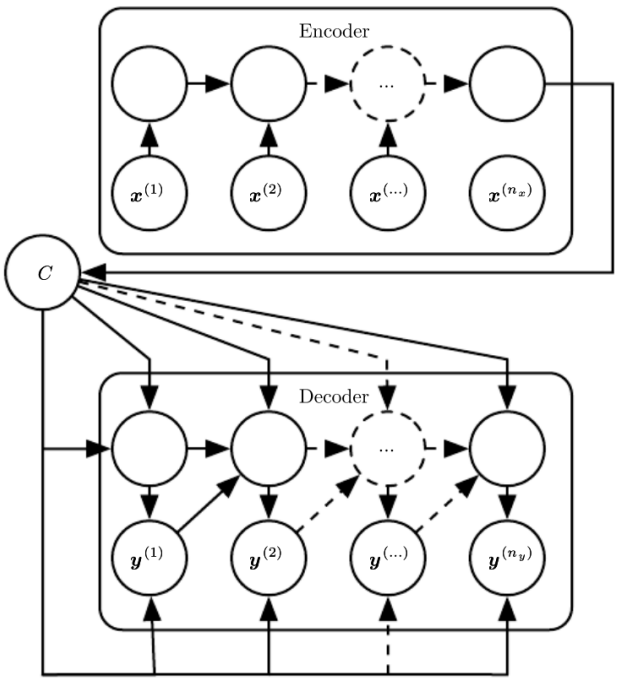
- informacja z przyszłości może zawierać istotną informację, np. dźwięki kolejnych fonemów (koartykulacja), znaczenie słowa w zdaniu może zależeć od słów występujących później
- wyjścia sieci w dowolnym momencie zależą od całej sekwencji wejściowej, możliwe tylko uczenie *offline*, gdy cała sekwencja jest znana
- zastosowanie: rozpoznawanie mowy (Graves, 2008, 2009), bioinformatyka (Baldi, 1999), rozpoznawanie pisma (Liwicki, Marcus, 2007)
- dla obrazów 2D możliwe zastosowanie sieci 4 kierunkowych (lewo, prawo, góra, dół), co pozwala modelować odległe relacje, jednak nie są one tak wydajne w tych zastosowaniach jak CNN

Model Encoder-Decoder

Transformacja sekwencji w inną sekwencję

$\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n_x)} \Rightarrow \mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(n_y)}$





Sutskever et al (2014). "Sequence to Sequence Learning with Neural Networks", NIPS.
K Cho et al (2014). "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation", EMNLP.

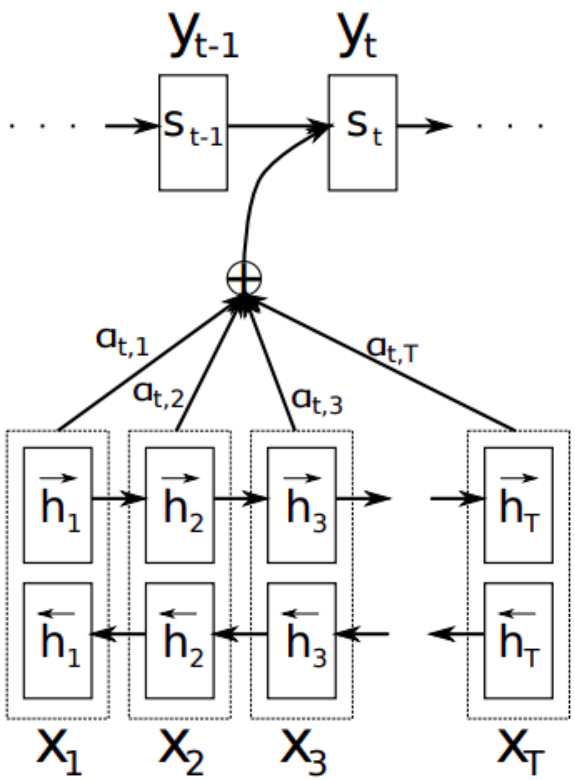
Model Encoder-Decoder

seq2seq (Sutskever, 2014) to dwie sieci RNN uczone jednocześnie:

- **(en)koder** (reader, input) przetwarza sekwencję wejściową generując kontekst **c** o stałej wielkości
- **dekoder** (writer, output) zależnie od kontekstu **c** generuje sekwencję wyjściową
- wielkość kontekstu **c** ogranicza zdolność sieci do reprezentacji dalekich w czasie relacji

Model seq2seq z systemem uwagi (Bahdanau, 2015)

Model RNN enkoder-dekoder z **mechanizmem uwagi** (*attention*) umożliwia dekoderowi „skupić uwagę” na wybranych elementach sekwencji wejściowej w różnych momentach i decydować o ich wpływie na generowane wyjście



Sieć modeluje prawdopodobieństwo

$$p(\mathbf{y}_i|\mathbf{y}_1,\ldots,\mathbf{y}_{i-1},\mathbf{X}) = g(\mathbf{y}_{i-1},\mathbf{s}_i,\mathbf{c}_i)$$

$\mathbf{s}_i$ to stany ukryte dekodera RNN

$$\mathbf{s}_i = f(\mathbf{y}_{i-1},\mathbf{s}_{i-1},\mathbf{c}_i)$$

$\mathbf{c}_i$ to wektor kontekstu, który jest sumą ważoną stanów enkodera $\mathbf{h}_j$

$$\mathbf{c}_i = \sum_{j=1}^{T} a_{ij}\mathbf{h}_j$$

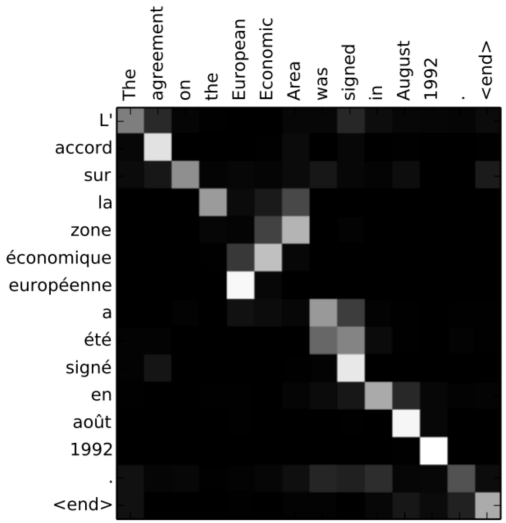
D. Bahdanau, et al (2016). Ńeural Machine Translation by Jointly Learning to Align and Translate”.

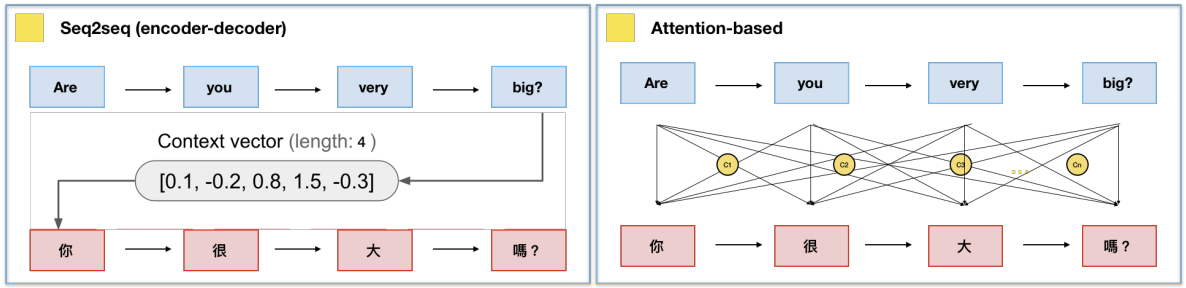
Model seq2seq z systemem uwagi (Bahdanau, 2015)

- **współczynniki uwagi** *attention score* a_{ij} są unormowane operacją softmax
$$a_{ij} = \frac{\exp e_{ij}}{\sum_{k=1}^{T} \exp e_{ik}}$$
- **model dopasowania** (alignment model) e_{ij} jest funkcją, która określa jak dobrze stan dekodera $\mathbf{s}_{i-1}$ pasuje do stanu enkodera $\mathbf{h}_j$
$$e_{ij} = f(\mathbf{s}_{i-1}, \mathbf{h}_j)$$
- model dopasowania może być prostą funkcją liniową, ale może też być złożoną siecią neuronową, np. MLP, trenowaną razem z siecią enkoder-dekoder, np.:
$$e_{ij} = \mathbf{w}_{ij}^T [\mathbf{s}_{i-1}; \mathbf{h}_j]$$

D. Bahdanau, et al (2016). Ńeural Machine Translation by Jointly Learning to Align and Translate".

Model seq2seq z systemem uwagi (Bahdanau, 2015)

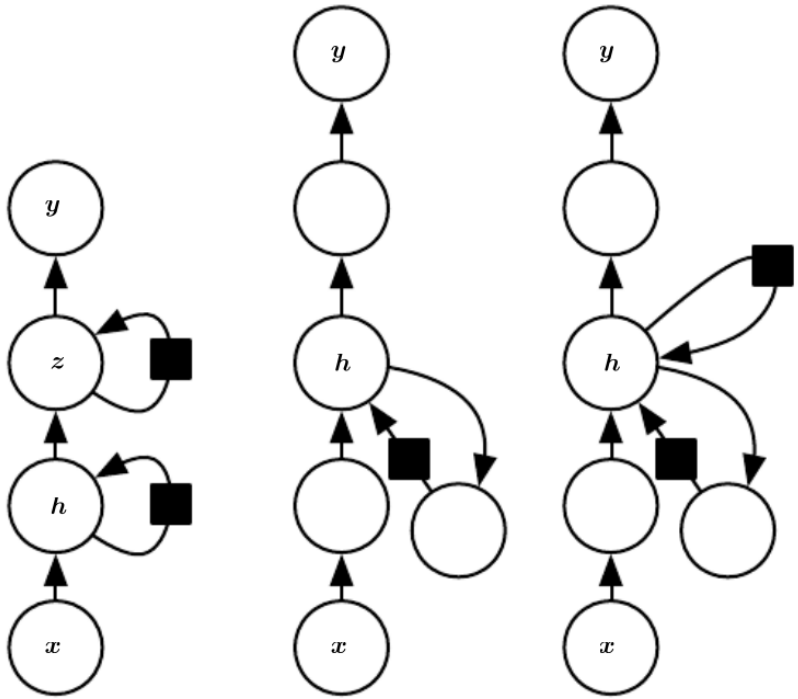




- współczynniki dopasowania a_{ij} są interpretowane jako prawdopodobieństwo, że dekoder w chwili i powinien skupić uwagę na stanie enkodera $\mathbf{h}_j$
- w trakcie generowania sekwencji dekoder może dynamicznie zmieniać uwagę na różne stany enkodera
- w przypadku tłumaczenia maszynowego dekoder skupia uwagę na różnych słowach w zdaniu źródłowym

D. Bahdanau, et al (2016). Ńeural Machine Translation by Jointly Learning to Align and Translate".

Deep RNN

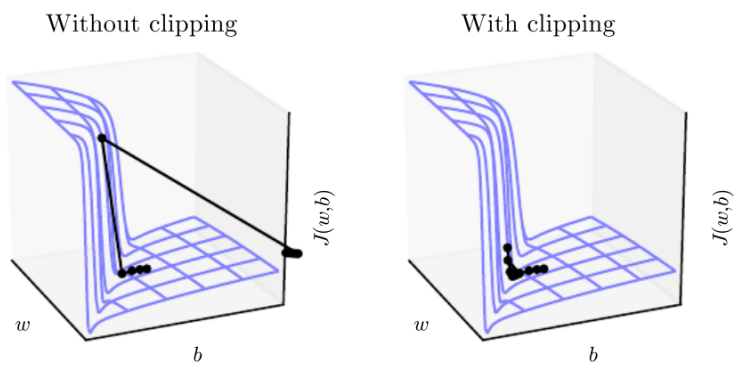


Wielowarstwowe RNN - kolejne warstwy tworzą hierarchię reprezentacji (Graves, 2013)

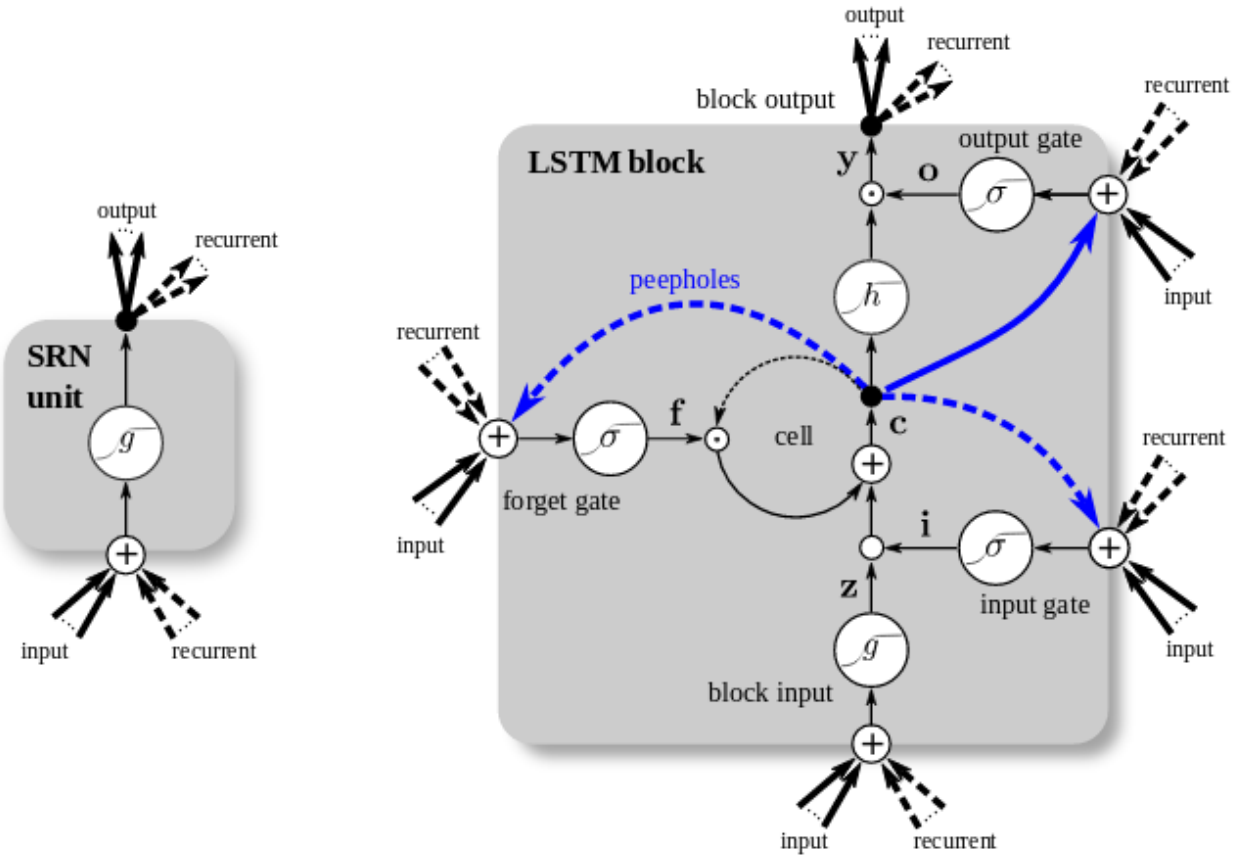
Goodfellow. Deep Learning, 2016

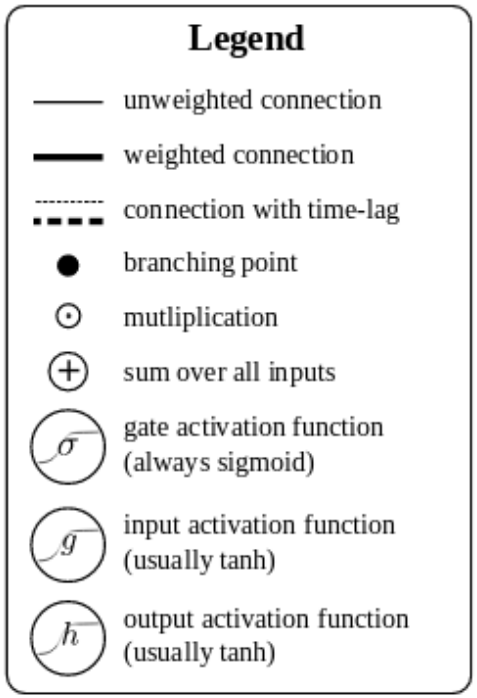
Problemy uczenia głębokich sieci RNN

- znikający gradient dla odległych w czasie relacji (zanika wykładniczo z czasem), rozwiązaniem LSTM
- niestabilność uczenia - eksplodujący gradient, rozwiązanie: przycinanie gradientu i regularyzacja
- **gradient clipping** - ograniczenie wartości gradientu dla wszystkich wymiarów w mini-batchu, może zmienić kierunek spadku gradientu
- **norm clipping**: jeśli $||g|| > v$ to $g \leftarrow \frac{gv}{||g||}$



Long Short-Term Memory





Sepp Hochreiter, Jürgen Schmidhuber, „Long short-term memory", 1997

LSTM

kandydat nowego stanu

$$\mathbf{z}^t = g\left(\mathbf{W}_z\mathbf{x}^t + \mathbf{R}_z\mathbf{y}^{t-1} + \mathbf{b}_z\right)$$

bramka wejściowa

$$\mathbf{i}^t = \sigma\left(\mathbf{W}_i\mathbf{x}^t + \mathbf{R}_i\mathbf{y}^{t-1} + \mathbf{b}_i\right)$$

bramka zapominania

$$\mathbf{f}^t = \sigma\left(\mathbf{W}_f\mathbf{x}^t + \mathbf{R}_f\mathbf{y}^{t-1} + \mathbf{b}_f\right)$$

stan wewnętrzny

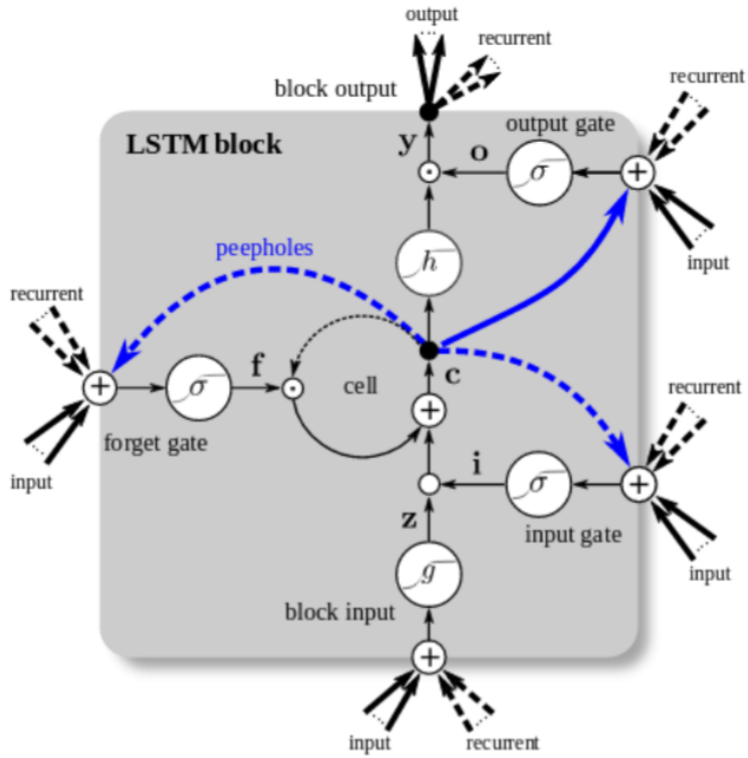
$$\mathbf{c}^t = \mathbf{i}^t \odot \mathbf{z}^t + \mathbf{f}^t \odot \mathbf{c}^{t-1}$$

bramka wyjściowa

$$\mathbf{o}^t = \sigma\left(\mathbf{W}_o\mathbf{x}^t + \mathbf{R}_o\mathbf{y}^{t-1} + \mathbf{b}_o\right)$$

sygnał wyjściowy

$$\mathbf{y}^t = \mathbf{o}^t \odot h\left(\mathbf{c}^t\right)$$



Sepp Hochreiter, Jürgen Schmidhuber, „Long short-term memory", 1997

LSTM

- LSTM niweluje wpływ zanikania gradientu, dzięki czemu pozwala mapować odległe w czasie relacje
- wprowadza dodatkową wewnętrzną jednostkę stanu $\mathbf{c}$
- stan $\mathbf{c}^t$ zależy w sposób liniowy od stanu poprzedniego $\mathbf{c}^{t-1}$ (liniowa rekurencja), więc nawet przy bardzo długich rekurencjach gradient ma szansę nie zaniknąć (*long-term memory*)
- iloczyn Hadamanda $\mathbf{x} \odot \sigma(\mathbf{y})$ (*po współrzędnych*), realizacja bramki wyłączającej/włączającej sygnał $\mathbf{x}$
- bramki wejściowa, wyjściowa i bramka zapominania sterują zapamiętywanymi informacjami, zależą od sygnału wejściowego i stanu poprzedniego
- funkcja różniczkowalna - można uczyć metodami spadku gradientu i wsteczną propagacją

LSTM

- bramka wejściowa ma wpływ na to, jaki sygnał zostanie zapisany w jednostce stanu $\mathbf{c}$
- bramka zapominania steruje wpływem poprzedniego stanu $\mathbf{c}^{t-1}$ na obecny stan $\mathbf{c}^{t}$
- bramka wejściowa i zapominania wpływają na to, co jest umieszczane i co jest usuwane ze stanu
- bramka wyjściowa steruje ilością informacji przepływającą ze stanu wewnętrznego do wyjścia jednostki, pozwala np. zachować na później informacje, które w tym momencie nie są istotne

Peephole LSTM

$$\mathbf{z}^t = g\left(\mathbf{W}_z \mathbf{x}^t + \mathbf{R}_z \mathbf{y}^{t-1} + \mathbf{b}_z\right)$$

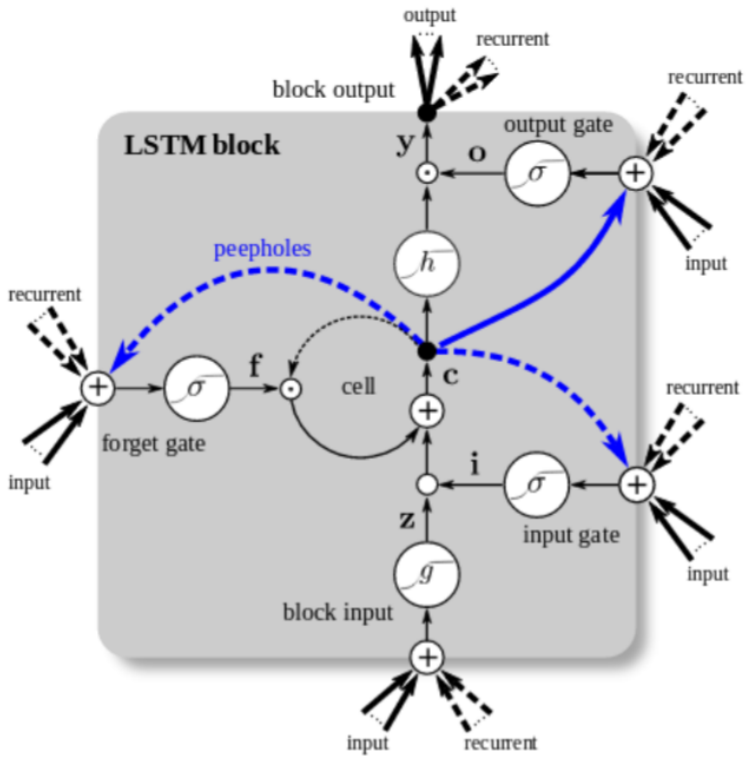
$$\mathbf{i}^t = \sigma\left(\mathbf{W}_i \mathbf{x}^t + \mathbf{R}_i \mathbf{y}^{t-1} + \mathbf{p}_i \odot \mathbf{c}^{t-1} + \mathbf{b}_i\right)$$

$$\mathbf{f}^t = \sigma\left(\mathbf{W}_f \mathbf{x}^t + \mathbf{R}_f \mathbf{y}^{t-1} + \mathbf{p}_f \odot \mathbf{c}^{t-1} + \mathbf{b}_f\right)$$

$$\mathbf{c}^t = \mathbf{i}^t \odot \mathbf{z}^t + \mathbf{f}^t \odot \mathbf{c}^{t-1}$$

$$\mathbf{o}^t = \sigma\left(\mathbf{W}_o \mathbf{x}^t + \mathbf{R}_o \mathbf{y}^{t-1} + \mathbf{p}_o \odot \mathbf{c}^t + \mathbf{b}_o\right)$$

$$\mathbf{y}^t = \mathbf{o}^t \odot h\left(\mathbf{c}^t\right)$$

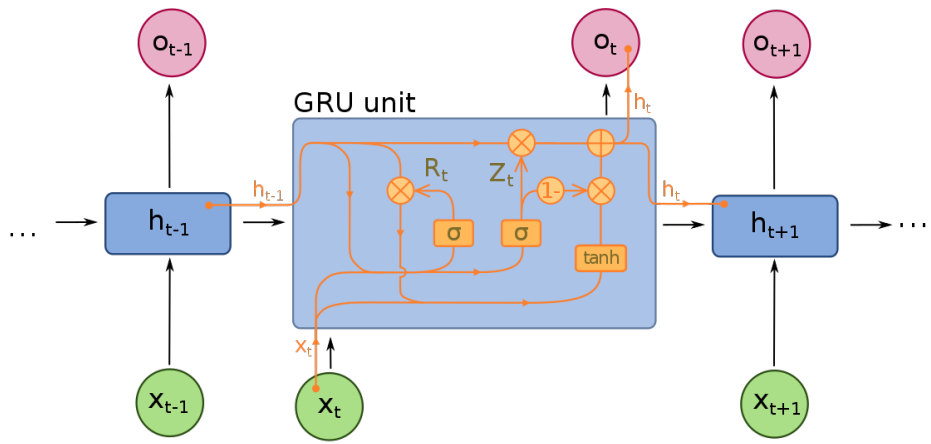


aktywacja bramek uzależniona dodatkowo od poprzedniego stanu $\mathbf{c}^{t-1}$ poprzez połączenia „podglądające” $\mathbf{p}_i$, $\mathbf{p}_o$, $\mathbf{p}_f$

Sepp Hochreiter, Jürgen Schmidhuber, „Long short-term memory”, 1997

Gated Recurrent Unit

- GRU (Kyunghyun Cho et al., 2014) brak jednostki wewnętrznej **c** i bramki wyjściowej
- bramki: resetu r_t i aktualizacji z_t
- mniej parametrów względem LSTM, porównywalna jakość wyników dla wielu problemów



Źródło: https://en.wikipedia.org/wiki/Recurrent_neural_network

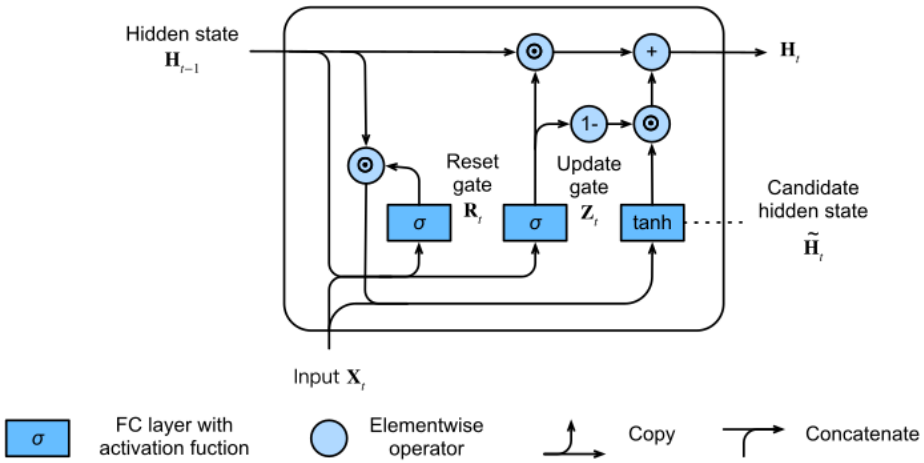
Gated Recurrent Unit

$$\mathbf{z}^t = \sigma\left(\mathbf{W}_z\mathbf{x}^t + \mathbf{U}_z\mathbf{h}^{t-1} + \mathbf{b}_z\right)$$

$$\mathbf{r}^t = \sigma\left(\mathbf{W}_r\mathbf{x}^t + \mathbf{U}_r\mathbf{h}^{t-1} + \mathbf{b}_r\right)$$

$$\tilde{\mathbf{h}}^t = \tanh\left(\mathbf{W}_h\mathbf{x}^t + \mathbf{U}_h\left(\mathbf{r}^t \circ \mathbf{h}^{t-1}\right) + \mathbf{b}_h\right)$$

$$\mathbf{h}^t = \left(1 - \mathbf{z}^t\right) \circ \mathbf{h}^{t-1} + \mathbf{z}^t \circ \tilde{\mathbf{h}}^t$$



- stan proponowany $\tilde{\mathbf{h}}^t$, dla $\mathbf{r}^t = 0$ sprowadza się do warstwy MLP
- bramka aktualizacji $\mathbf{z}^t$ określa stopień wpływu poprzedniego stanu $\mathbf{h}^{t-1}$ oraz obecnego stanu proponowanego $\tilde{\mathbf{h}}^t$
- bramka resetu odpowiada za relacje krótkie w czasie, a bramka aktualizacji - za długie

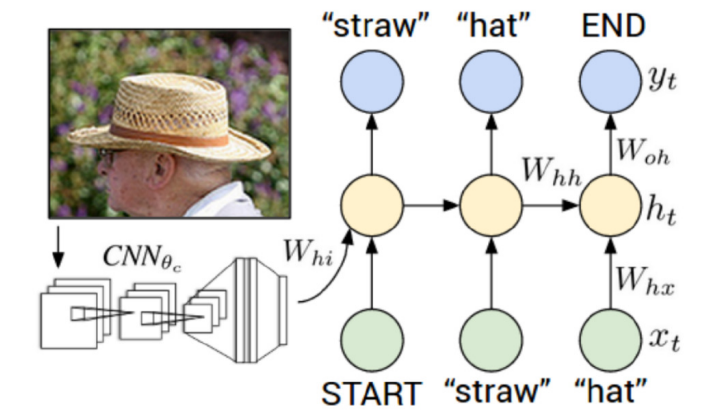
Źródło grafiki: https://d2l.ai/chapter_recurrent-modern/gru.html

Zastosowania: generator opisów zdjęć







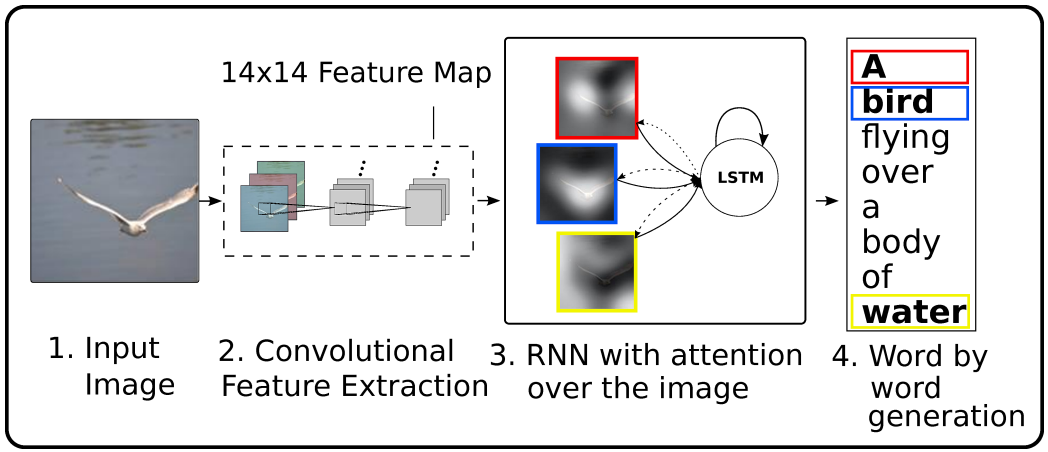


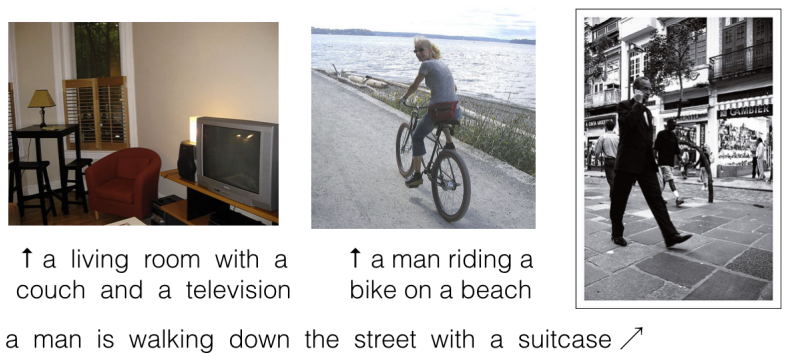
Multimodal Recurrent Neural Network ☞ NeuralTalk2

Karpathy A., Fei-Fei L. „Deep Visual-Semantic Alignments for Generating Image Descriptions", 2015

Zastosowania: generator opisów zdjęć

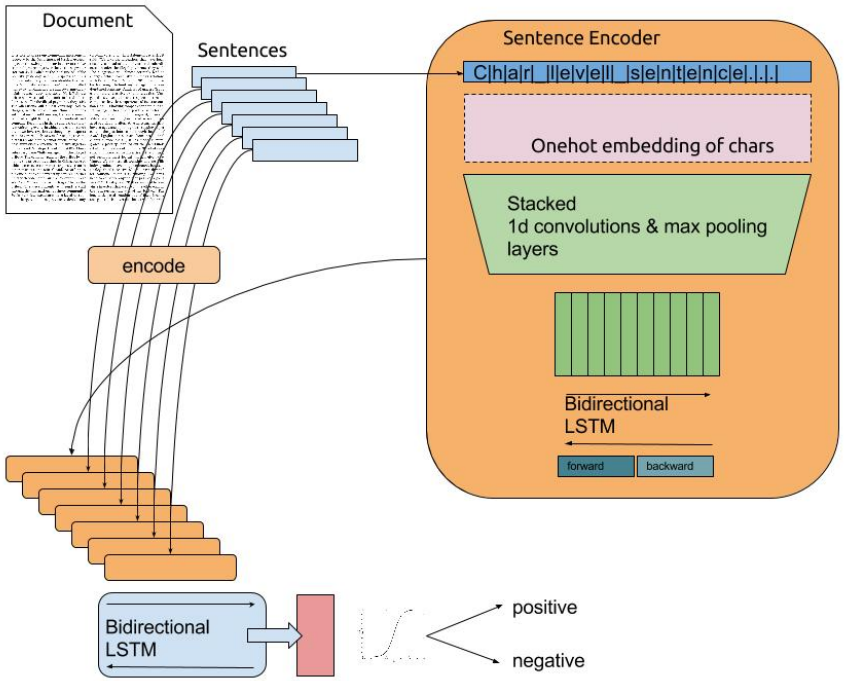
Show, attend and tell (Xu et al., 2015)





Mechanizm uwagi pozwala skupić się na wybranych fragmentach obrazu podczas generowania opisu

Zastosowania: analiza sentymentów



https://offbit.github.io/how-to-read/

Zastosowania: odpowiedzi na pytania

