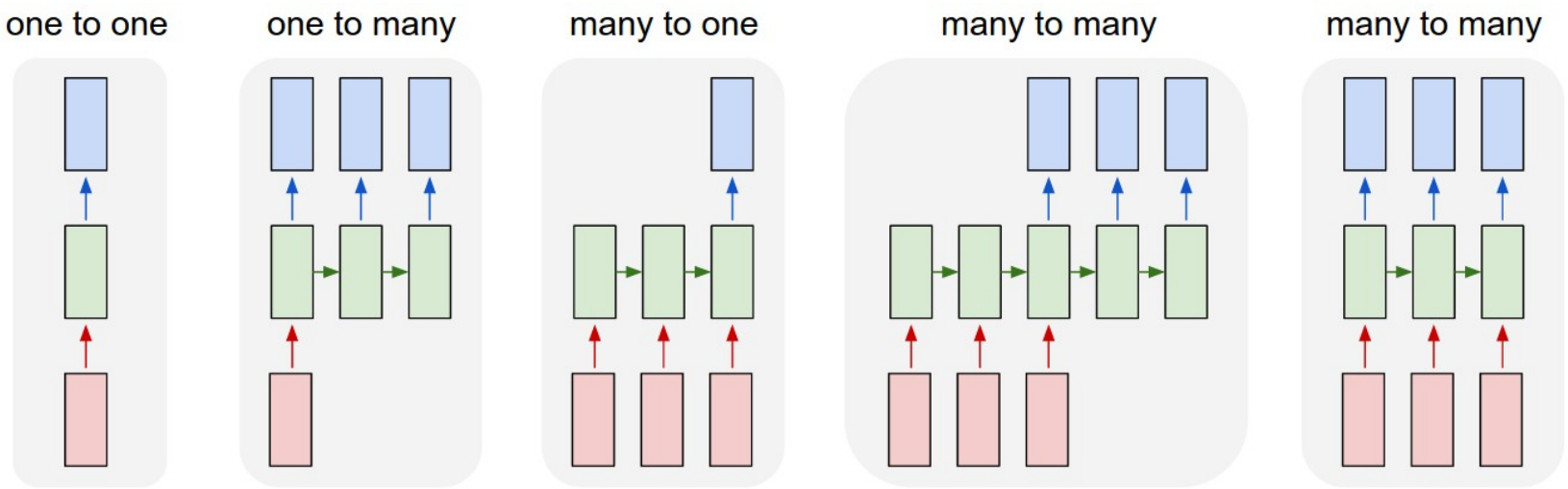
RNN

Sieci rekurencyjne i modelowanie sekwencji

Modelowanie sekwencji

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

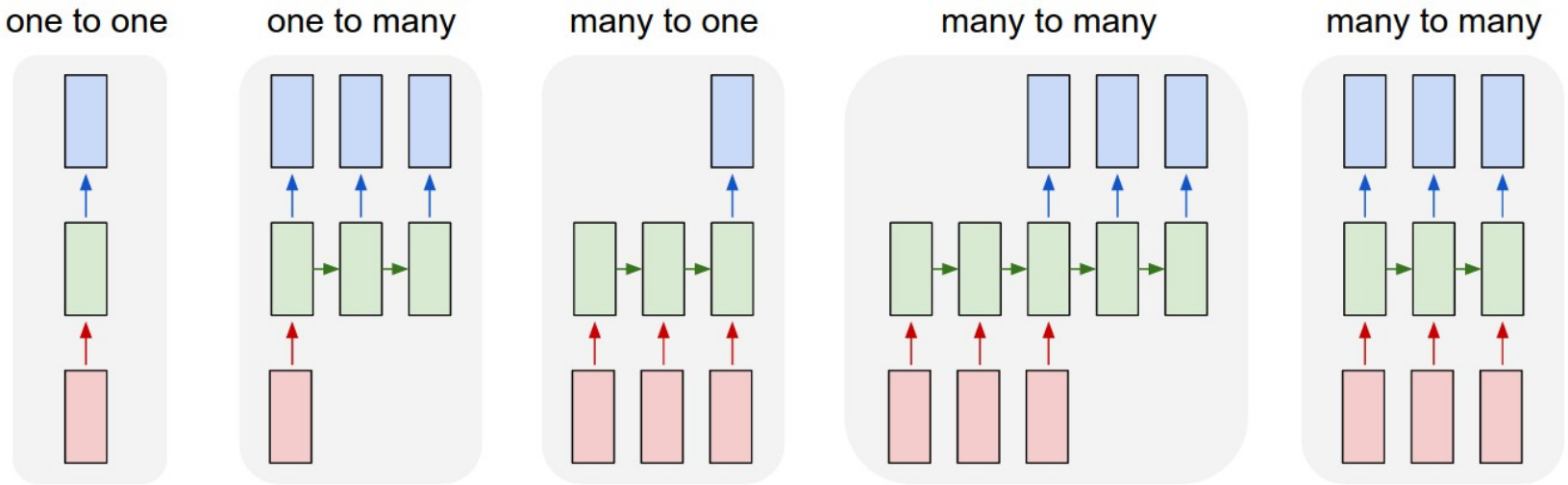
Modelowanie sekwencji



- *one-to-one*: pojedynczy sygnał wejściowy produkuje pojedynczy sygnał wyjściowy, brak informacji o sekwencji
- *one-to-many*: pojedyncze wejście generuje sekwencję na wyjściu, np. opis obrazów
- *many-to-one*: sekwencja wejściowa produkuje pojedynczy sygnał wyjściowy, np. klasyfikacja tekstów

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, rozpoznawanie 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

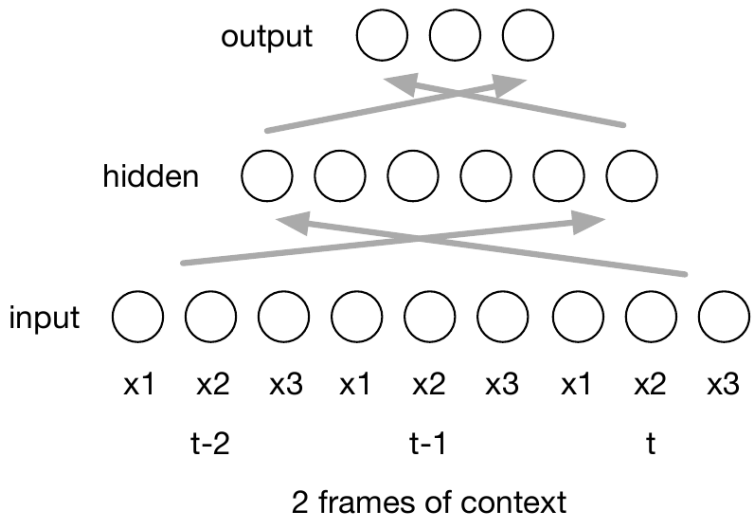
- MLP: dodanie **stałego kontekstu** do wejścia: np. 2 ramki z lewej i prawej (poprzednie i przyszłe wektory wejściowe)

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

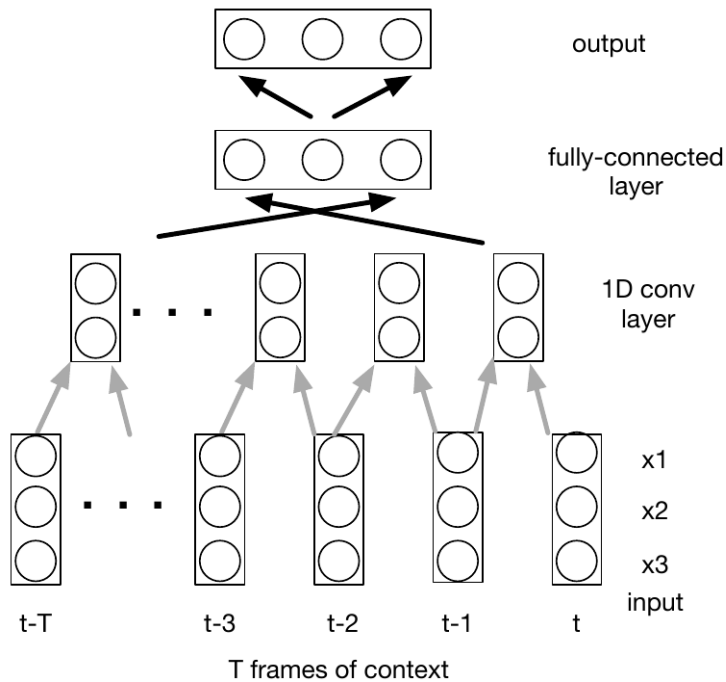
- CNN: splot względem wymiaru czasu, stały kontekst odpowiada wielkości efektywnego pola recepcyjnego w wymiarze czasu.
  TDNN (*time-delay neural network*) model ze splotem 1D w domenie czasu.

Uczenie sekwencji z MLP i CNN

MLP

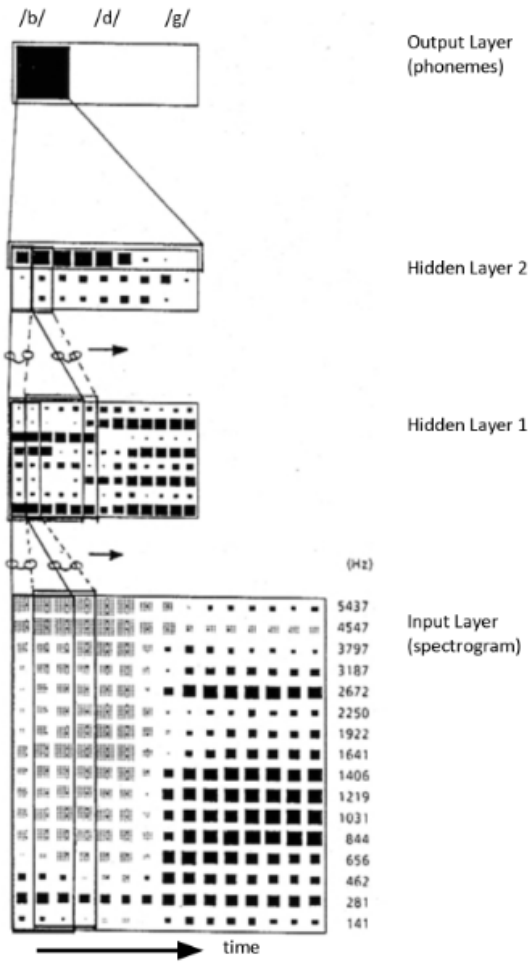


TDNN



Źródło grafik: Steve Renals, Machine Learning Practical — MLP Lecture 9

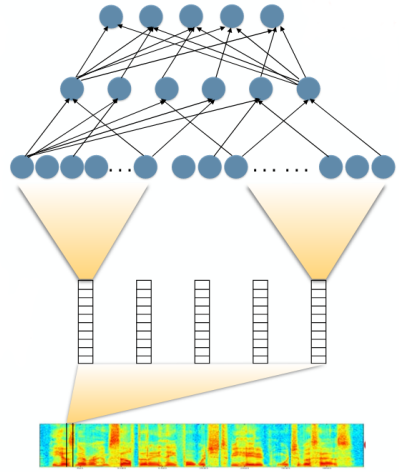
TDNN (Waibel, 1987)



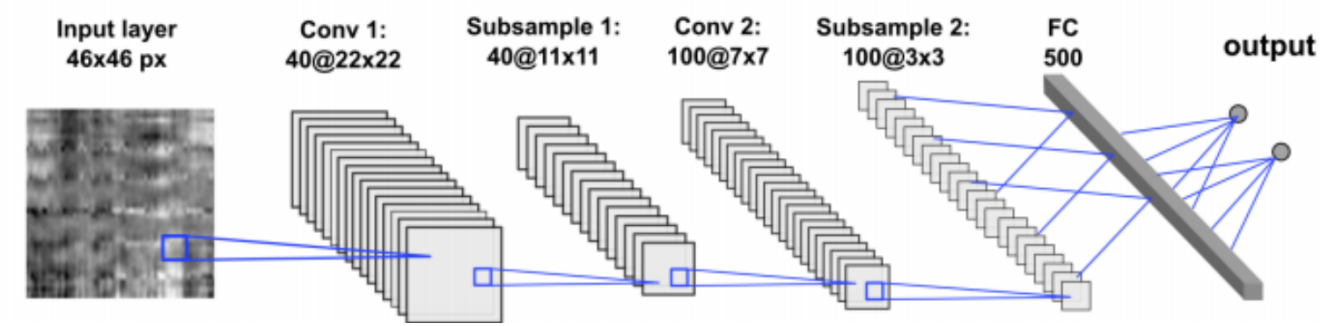
Alexander Waibel, Phoneme Recognition Using Time-Delay Neural Networks, 1987.

Przykłady

Model akustyczny, rozpoznawanie fonemów z nagrań audio



Analiza sygnałów EEG, CNN + widma EEG



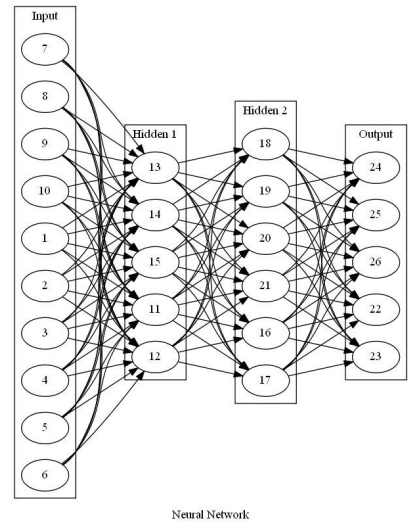
Nurse, E., Mashford, B. S., Yepes, A. J., Kiral-Kornek, I., Harrer, S., & Freestone, D. R. (2016). Decoding EEG and LFP Signals using Deep Learning: Heading TrueNorth

RNN - Sieci rekurencyjne

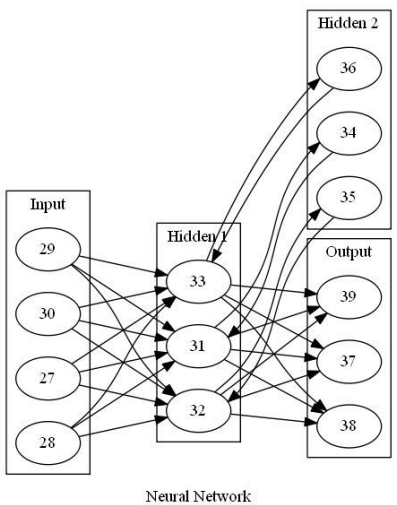
- **Sieci jednokierunkowe** używają jedynie kontekstu o skończonej długości, nie są w stanie wykryć zależności w dłuższych okresach czasu (*long-term*)

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

Sieć jednokierunkowa

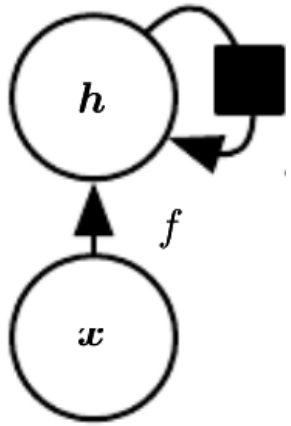


Sieć rekurencyjna Elmana



Stan ukryty jednostek RNN

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

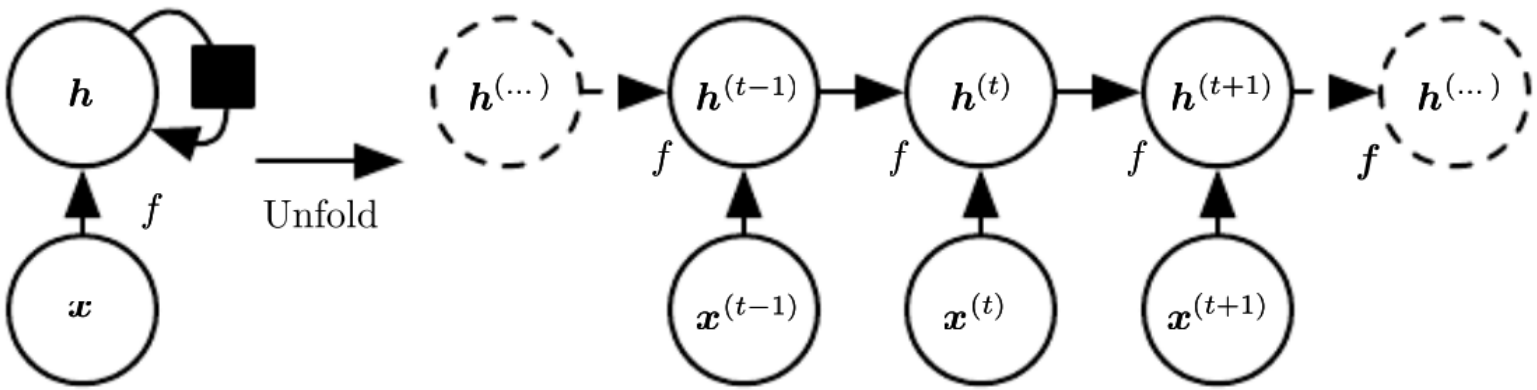


- neurony z połączeniami rekurencyjnymi działają jak pamięć - potencjalnie są w stanie modelować relacje występujące z dowolnie długim odstępem czasowym (nieskończony kontekst)

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

- 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 zerami

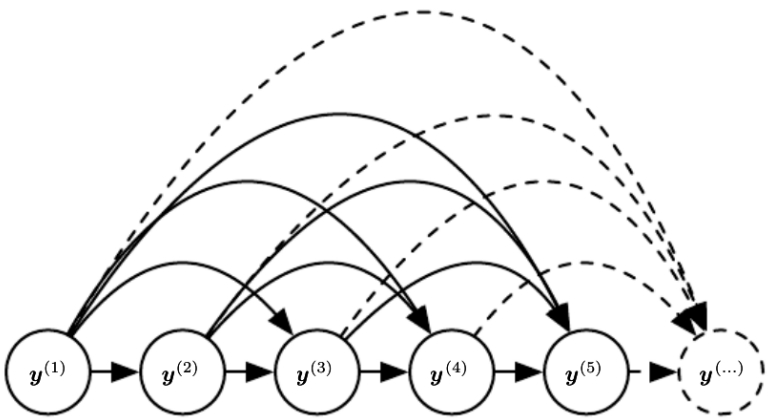
Rozwinięcie rekurencji w czasie



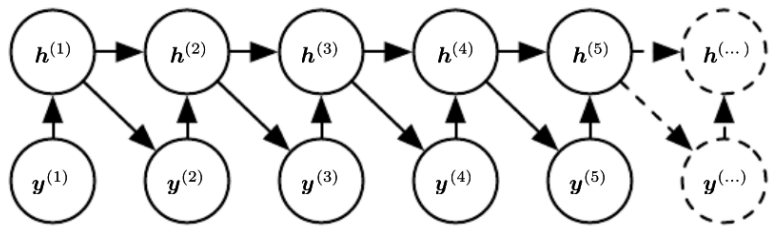
Rozwinięcie sieci RNN do postaci sieci jednokierunkowej, gdzie wagi połączeń są współdzielone

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

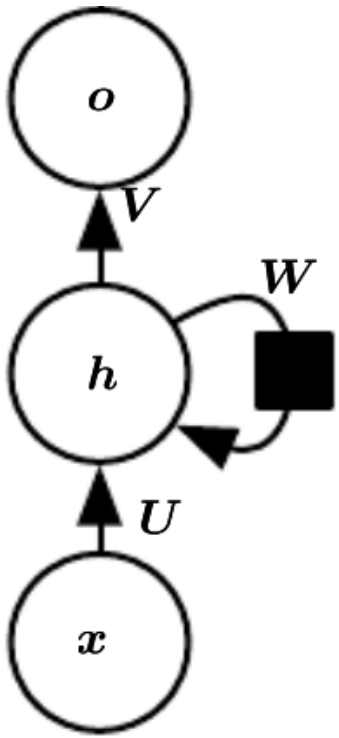


pełny model sekwencji, ilość parametrów $O(k^\tau)$



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

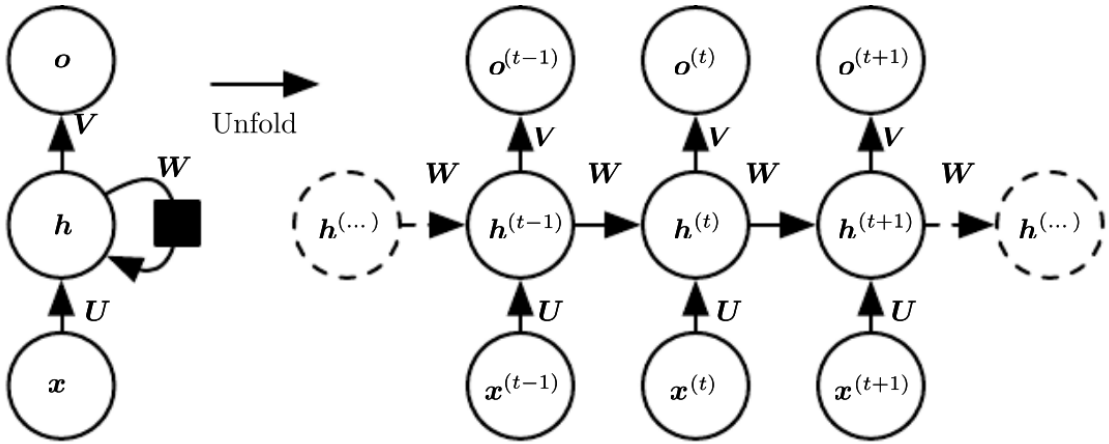
$\mathbf{U} \in \mathbb{R}^{d \times k}, \quad \mathbf{W} \in \mathbb{R}^{k \times k}$

d ilość wejść sieci

k ilość neuronów w warstwie ukrytej

Źródło grafiki: Goodfelow, Deep Learning, 2016

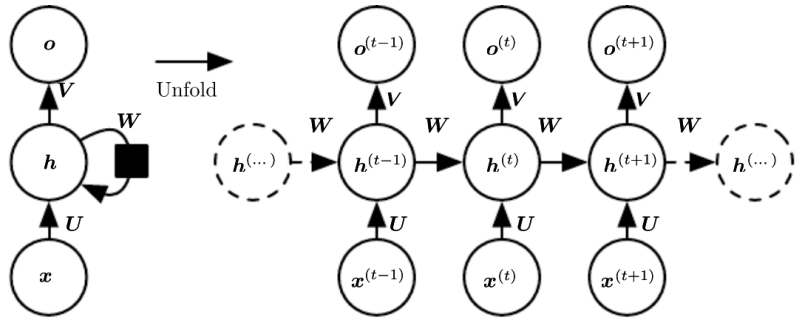
Wsteczna propagacja w czasie



- Rozwinięta sieć odpowiada głębokiej sieci jednokierunkowej ze współdzielonymi wagami $\mathbf{W}, \mathbf{V}, \mathbf{U}$
- BPTT *backpropagation through time* - trening za pomocą wstecznej propagacji na sieci „rozwiniętej w czasie", odbywa się analogicznie jak w jednokierunkowych sieciach.

Źródło grafiki: Goodfelow, Deep Learning, 2016

Wsteczna propagacja w czasie



Funkcja kosztu dla całej sekwencji:

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

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

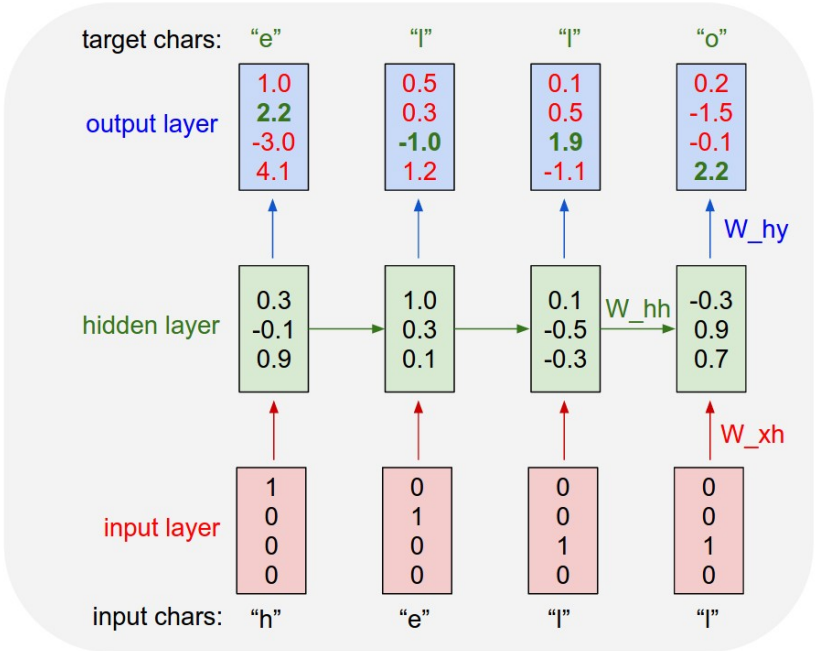
$$\partial_w L = \sum_{t=1}^{\tau} \partial_w L\left(y^{(t)}, o^{(t)}\right)$$

Aktualizacje wag są akumulowane dla sygnału błędu propagowanego w kolejnych krokach czasu

Źródło grafiki: Goodfelow, Deep Learning, 2016

Przykład: CharRNN

- model języka oparty na znakach
- wejście to sekwencja znaków, każdy znak to wektor o długości równej ilości znaków w alfabecie
- RNN modeluje rozkład prawdopodobieństwa wystąpienia następnego znaku w sekwencji $f(x(t)) \rightarrow x(t+1)$

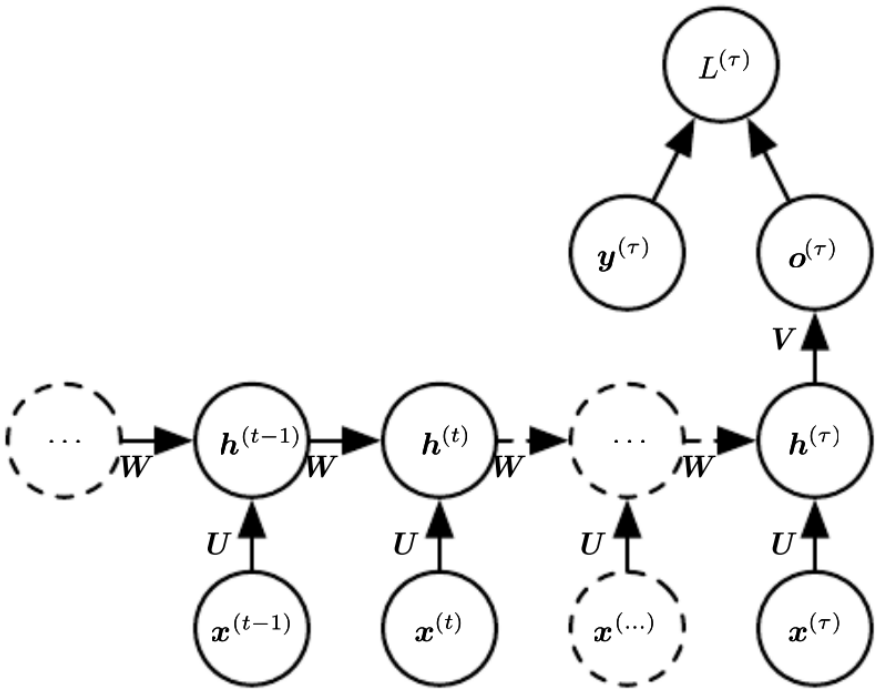


☞ The Unreasonable Effectiveness of Recurrent Neural Networks

Wsteczna propagacja w czasie

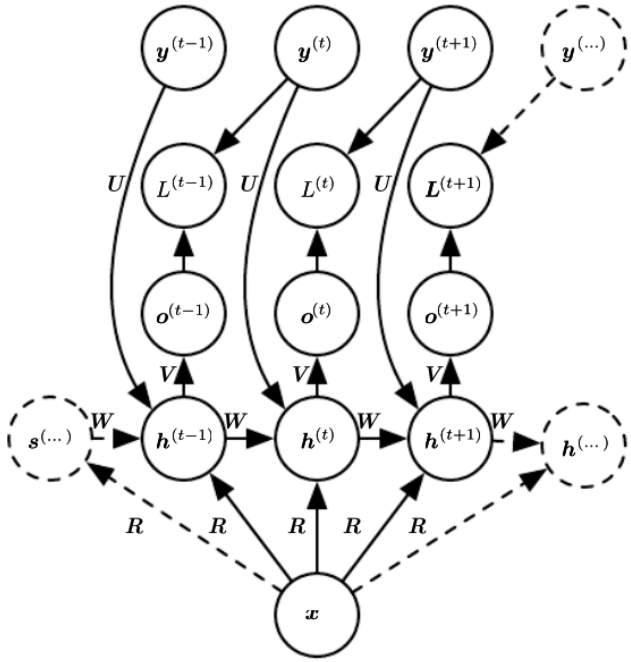
- Rozwinięta w czasie sieć tworzy głęboką sieć neuronową ze wszystkimi tego konsekwencjami (zanikający gradient dla relacji odległych w czasie, rzadziej - wybuchający gradient)
- Obliczenia dla długich sekwencji są kosztowne
- **Truncated BPTT** - ograniczenie długości sekwencji do stałej wartości (np. 20)

Pojedyncze wyjście na końcu sekwencji



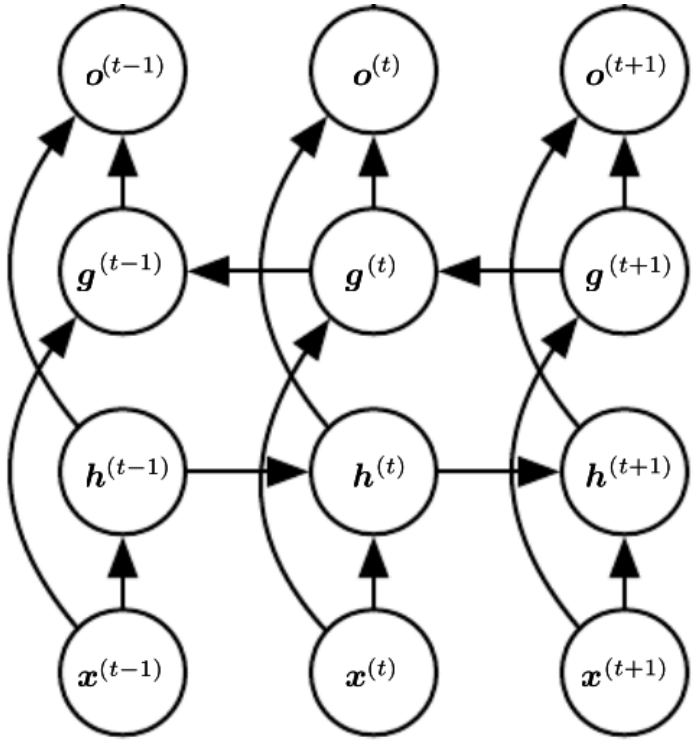
Sieć produkuje wyjście po obejrzeniu całej sekwencji,
np. klasyfikacja wpisów w portalach społecznościowych
(tzw. analiza sentymentów)

Pojedyncze wejście generujące sekwencję



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

Dwukierunkowa sieć RNN



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

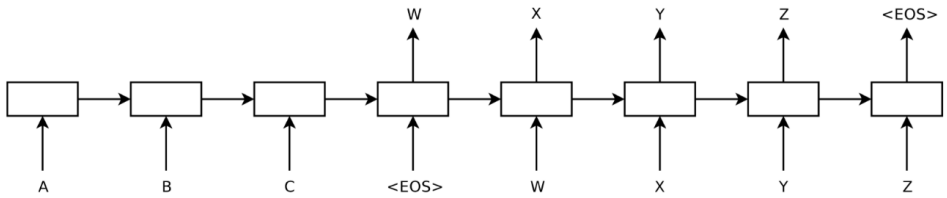
Schuster, Mike, and Kuldip K. Paliwal. "Bidirectional recurrent neural networks.Śignal Processing, 1997

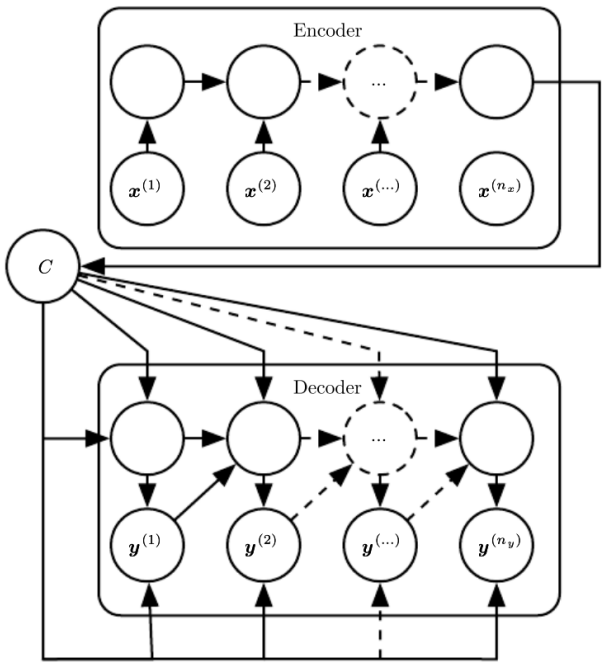
Dwukierunkowa sieć RNN

- informacja z przyszłości może zawierać istotną informację, np. dzwięki kolejnych fonemów (koartykulacja), znaczenie słów w zdaniu
- wyjścia sieci w dowolnym momencie zależą od całej sekwencji wejściowej
- zastosowanie: rozpoznawanie mowy (Graves, 2008, 2009), bioinformaryka (Baldi, 1999), rozpoznawanie pisma (Liwicki, Marcus, 2007)
- dla obrazów 2D możliwe zastosowanie sieci 4 kierunkowych, co pozwala modelować odległe relacje, jednak nie są one tak wydajne w tych zastosowaniach jak CNN

Model Encoder-Decoder

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





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.

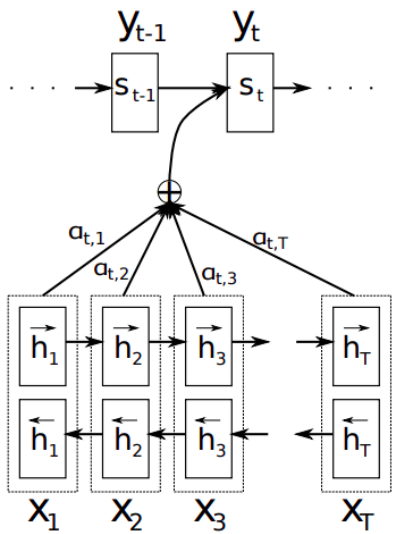
Model Encoder-Decoder

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

- **enkoder** (reader, input) przetwarza sekwencję wejściową generując kontekst C
- **dekoder** (writer, output) zależnie od kontekstu C generuje sekwencję wyjściową
- wielkość kontekstu C może ograniczyć zdolność sieci do reprezentacji dalekich w czasie relacji
- dodanie uwagi (*attention model*, Bahdanan, 2015) - uwzględnienie w C więcej informacji z enkodera, które skalują wyjścia dekodera

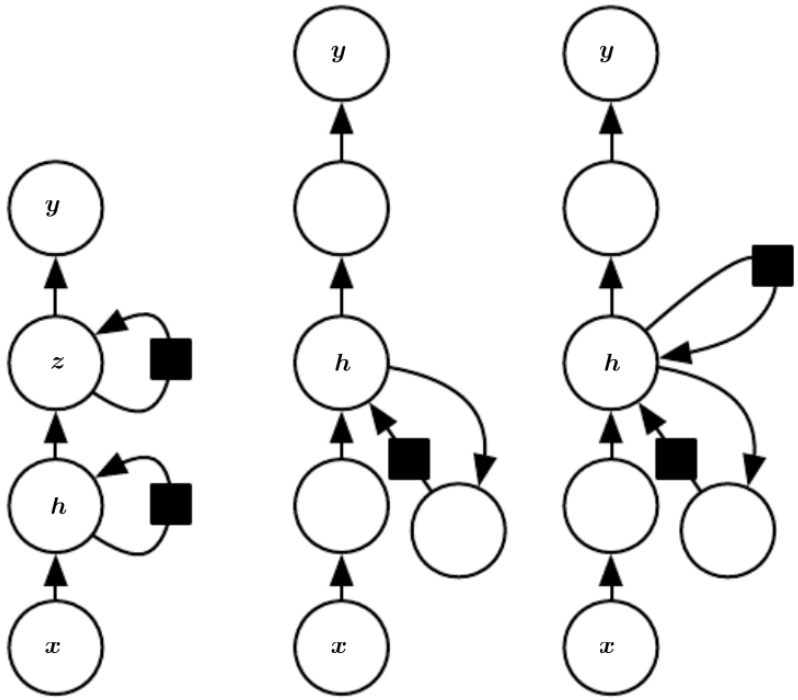
Model Encoder-Decoder a atencją

- **enkoder** dwukierunkowa sieć RNN, widzi całą sekwencję, genetuje wektor anotacji $\mathbf{h}_i$
- kontekst c_i jest sumą ważoną wszystkich anotacji
- mechanizm atencji waży wpływ kontekstu w wybranym momencie sekwencji wejściowej
- **dekoder** emuluje przeszukiwanie sekwencji wejściowej podczas transkrypcj˙



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

Deep RNN

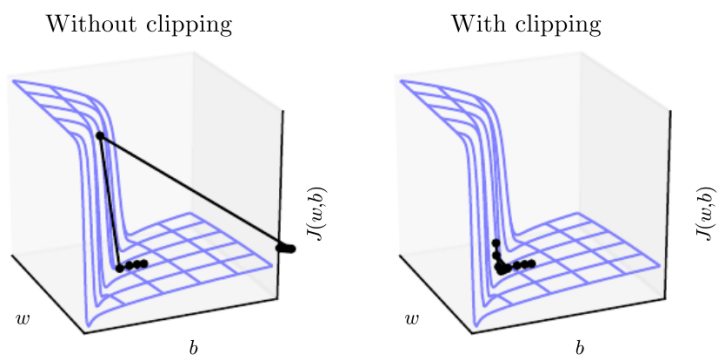


Wielowarstwowe RNN - kolejne warstwy tworzą hierarchię reprezentacji (Graves, 2013)

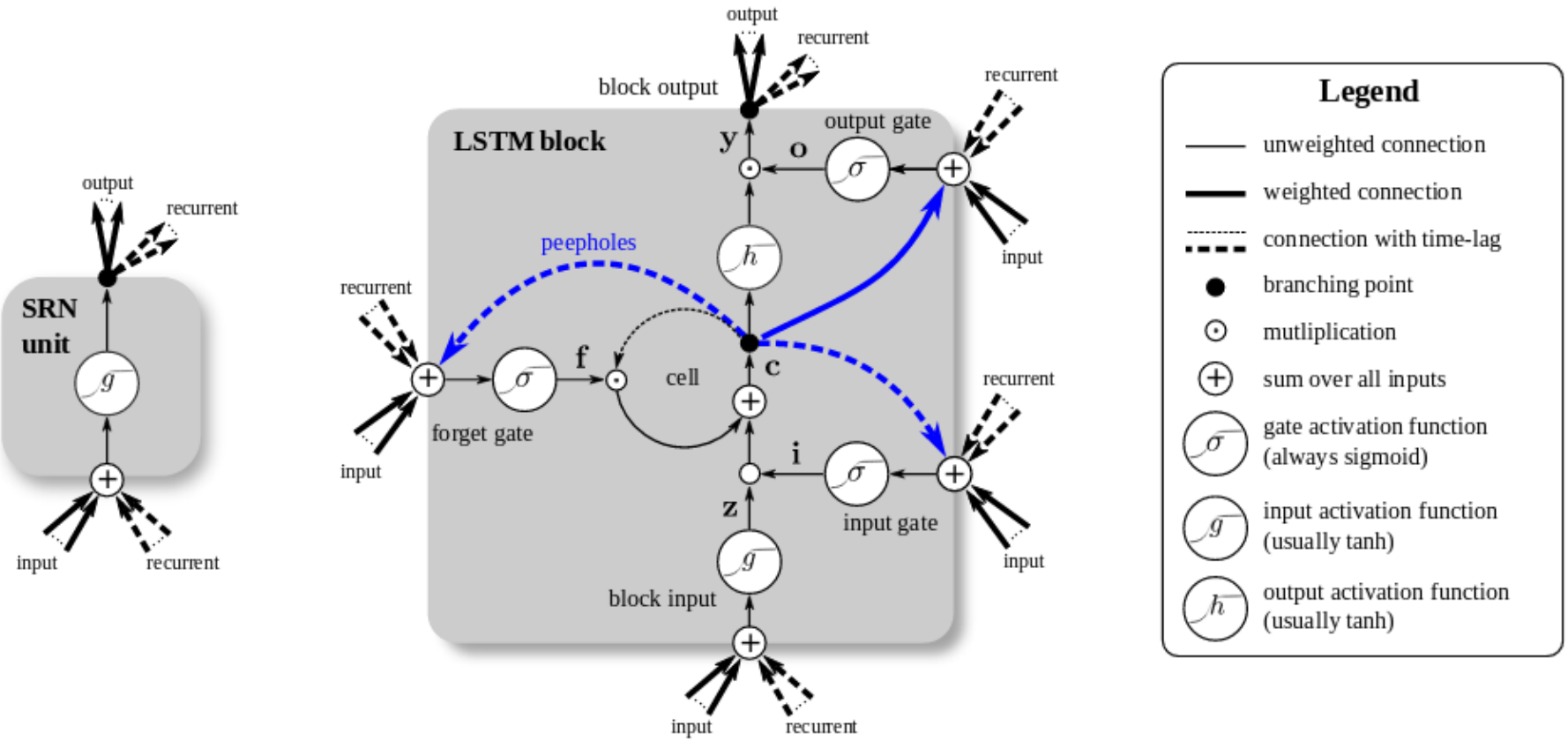
Goodfelow. Deep Learning, 2016

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 $g \leftarrow \frac{gv}{||g||}$



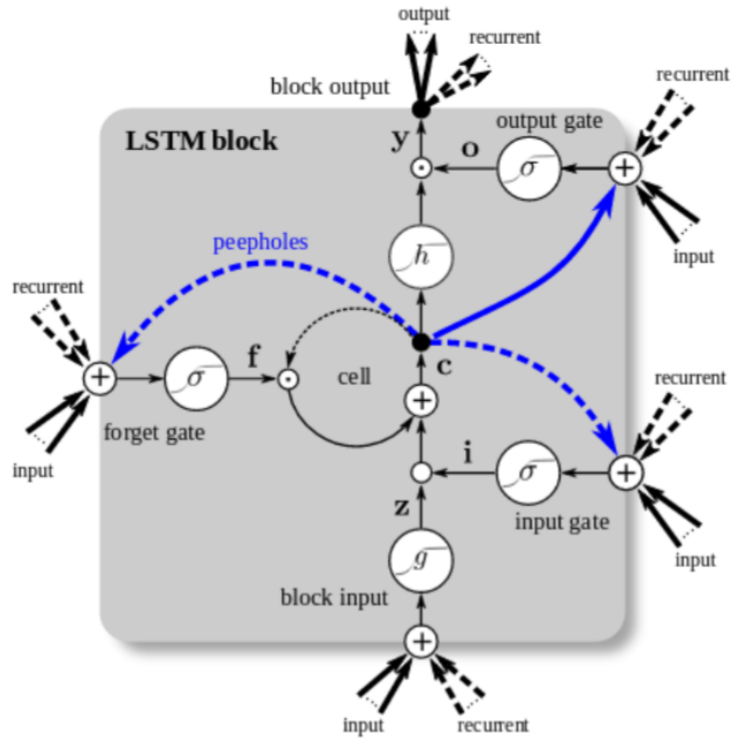
Long Short-Term Memory



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

LSTM

$$\mathbf{z}^t = g\left(\mathbf{W}_z \mathbf{x}^t + \mathbf{R}_z \mathbf{y}^{t-1} + \mathbf{b}_z\right)$$
$$\mathbf{i}^t = \sigma\left(\mathbf{W}_i \mathbf{x}^t + \mathbf{R}_i \mathbf{y}^{t-1} + \mathbf{b}_i\right)$$
$$\mathbf{f}^t = \sigma\left(\mathbf{W}_f \mathbf{x}^t + \mathbf{R}_f \mathbf{y}^{t-1} + \mathbf{b}_f\right)$$
$$\mathbf{c}^t = \mathbf{i}^t \odot \mathbf{z}^t + \mathbf{f}^t \odot \mathbf{c}^{t-1}$$
$$\mathbf{o}^t = \sigma\left(\mathbf{W}_o \mathbf{x}^t + \mathbf{R}_o \mathbf{y}^{t-1} + \mathbf{b}_o\right)$$
$$\mathbf{y}^t = \mathbf{o}^t \odot h\left(\mathbf{c}^t\right)$$



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

LSTM

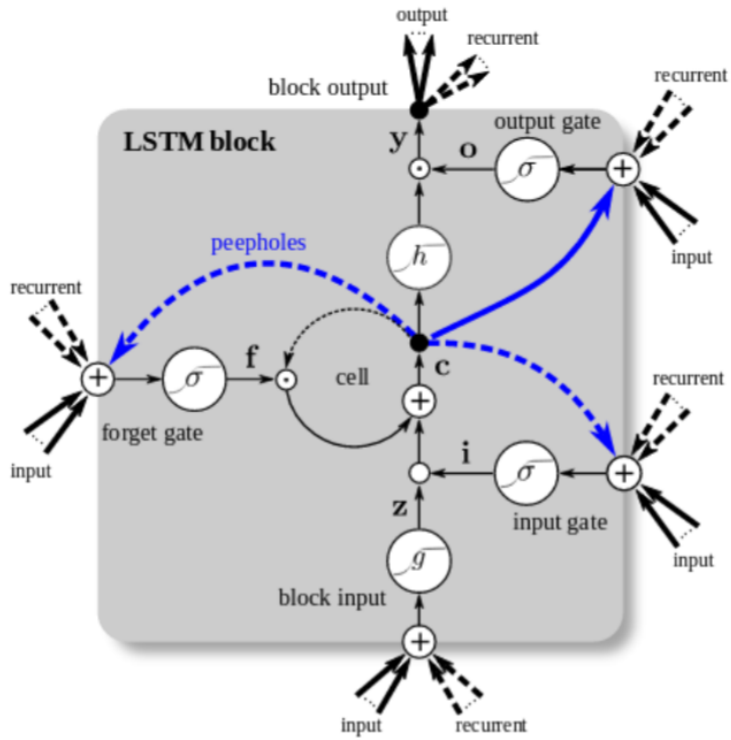
- LSTM niweluje wpływ zanikania gradientu, dzięki czemu pozwala mapować odległe w czasie relacje
- wprowadza dodatkową wewnętrzną jednostkę stanu $\mathbf{c}$
- stan $\mathbf{c}^t$ zależy w sposób liniowy od stanu poprzedniego $\mathbf{c}^{t-1}$ (liniowa rekurencja), więc nawet przy bardzo długich rekurencjach gradient ma szanse 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 umieszczane jest 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

$$\mathbf{z}^t = g\left(\mathbf{W}_z \mathbf{x}^t + \mathbf{R}_z \mathbf{y}^{t-1} + \mathbf{b}_z\right)$$
$$\mathbf{i}^t = \sigma\left(\mathbf{W}_i \mathbf{x}^t + \mathbf{R}_i \mathbf{y}^{t-1} + \mathbf{p}_i \odot \mathbf{c}^{t-1} + \mathbf{b}_i\right)$$
$$\mathbf{f}^t = \sigma\left(\mathbf{W}_f \mathbf{x}^t + \mathbf{R}_f \mathbf{y}^{t-1} + \mathbf{p}_f \odot \mathbf{c}^{t-1} + \mathbf{b}_f\right)$$
$$\mathbf{c}^t = \mathbf{i}^t \odot \mathbf{z}^t + \mathbf{f}^t \odot \mathbf{c}^{t-1}$$
$$\mathbf{o}^t = \sigma\left(\mathbf{W}_o \mathbf{x}^t + \mathbf{R}_o \mathbf{y}^{t-1} + \mathbf{p}_o \odot \mathbf{c}^t + \mathbf{b}_o\right)$$
$$\mathbf{y}^t = \mathbf{o}^t \odot h\left(\mathbf{c}^t\right)$$

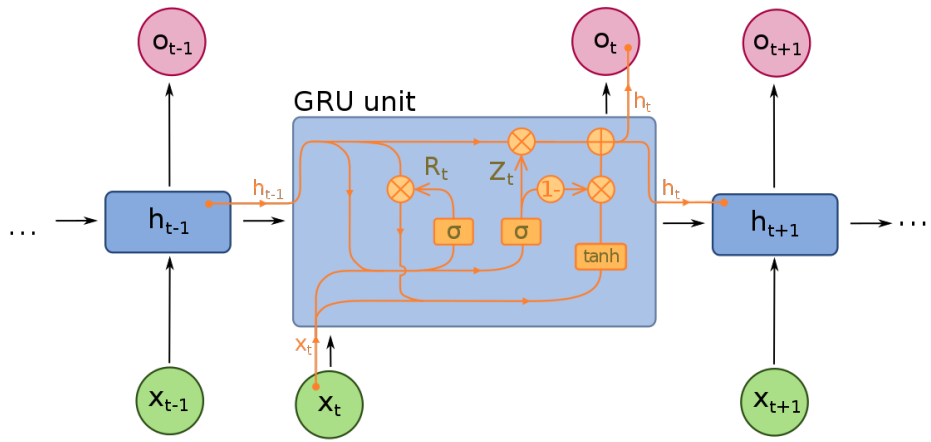


aktywacja bramek uzalezniona dodatkowo od poprzedniego stanu $\mathbf{c}^{t-1}$ poprzez połączenia „podglądające” $\mathbf{p}_i$, $\mathbf{p}_o$, $\mathbf{p}_f$

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

Gated Reccurent Unit

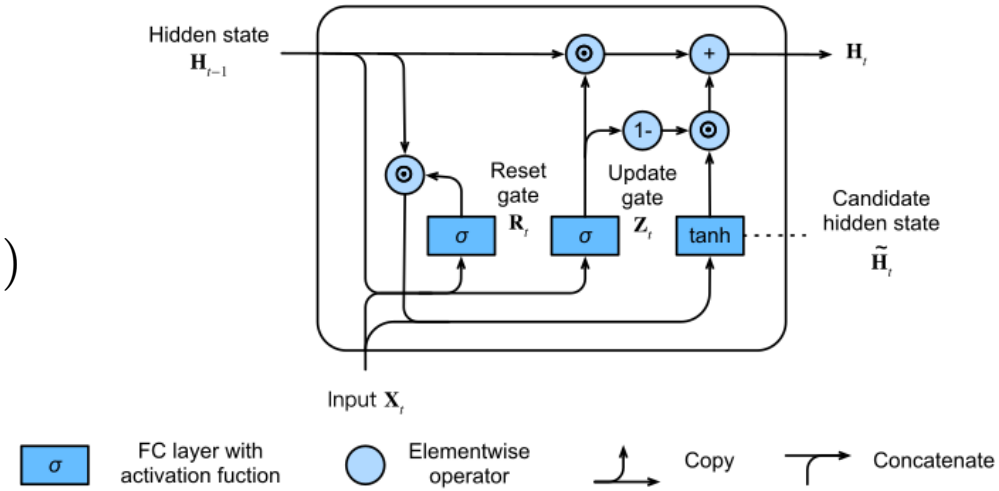
- GRU (Kyunghyun Cho et al., 2014) brak jednostki wewnętrznej **c** i bramki wyjściowej
- bramki: resetu r_t i aktualizacji z_t
- mniej parametrów względem LSTM, porównywalna jakość wyników dla wielu problemów



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

Gated Reccurent Unit

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



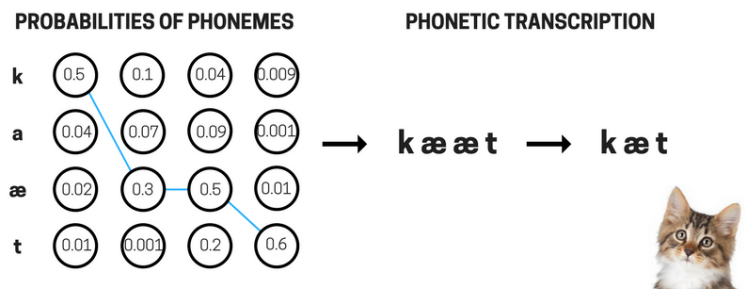
- stan proponowany $\tilde{\mathbf{h}}^t$, dla $\mathbf{r}^t = 0$ sprowadza się do warstwy MLP
- bramka aktualizacji $\mathbf{z}^t$ określa stopień wpływu poprzedniego stanu $\mathbf{h}^{t-1}$ oraz obecnego stanu proponowanego $\tilde{\mathbf{h}}^t$
- bramka resetu odpowiada za relacje krótkie w czasie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

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

Connectionist Temporal Classification

- metoda treningu sekwencji, gdy nie ma dostępnego dopasowania ramek (*aligement*), długość sekwencji wejściowej jest dłuższa od sekwencji wyjściowej, np. transkrypcja tekstu na podstawie nagrań audio
- wyjście softmax uzupełnione o dodatkową etykietę 'blank' $\mathcal{L}' = \mathcal{L} \cup \{blank\}$
- prawdopodobieństwo $P(\mathbf{y}'|\mathbf{x})$ zaobserwowania sekwencji etykiet (ścieżki) $\mathbf{y}'$ o długości T

$$P(\mathbf{y}'|\mathbf{x}) = \prod_{t=1}^{T} P(y'_t|\mathbf{x})$$



Connectionist Temporal Classification

CTC poszukuje sekwencji etykiet o największej wiarygodności $P(\mathbf{y}|\mathbf{x})$ marginalizując po wszystkich możliwych ścieżkach $\mathbf{y}'$ z blankiem

$$p(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{y}' \in \mathcal{M}^{-}1(\mathbf{y})} P(\mathbf{y}'|\mathbf{x})$$

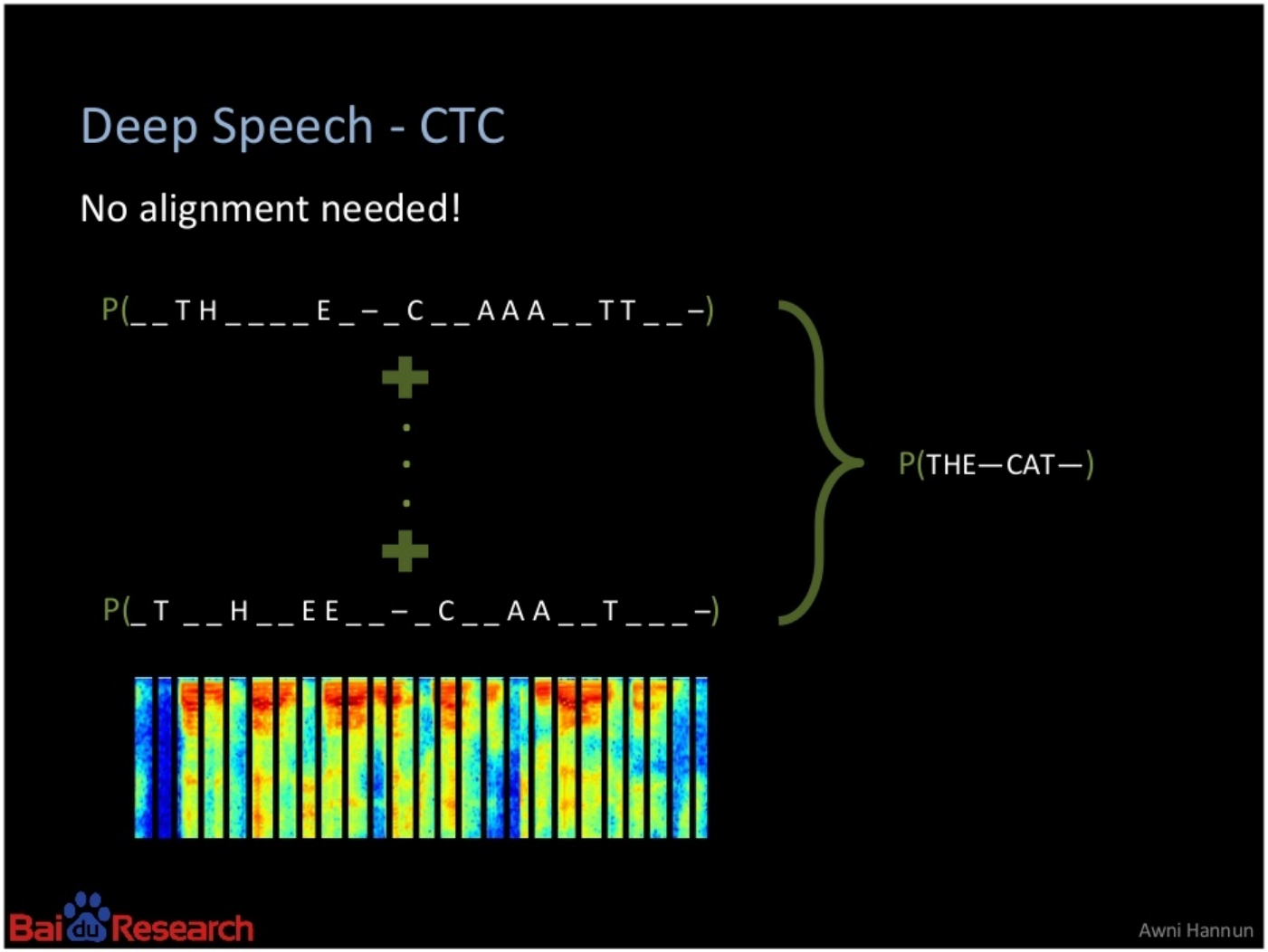
gdzie $\mathcal{M}$ mapuje sekwencję etykiet o długości T do sekwencji krótszej poprzez usunięcie wszystkich blanków i powtarzających się etykiet $\mathcal{M}(a - ab-) = \mathcal{M}(-aa - -abb) = aab$

Funkcja kosztu

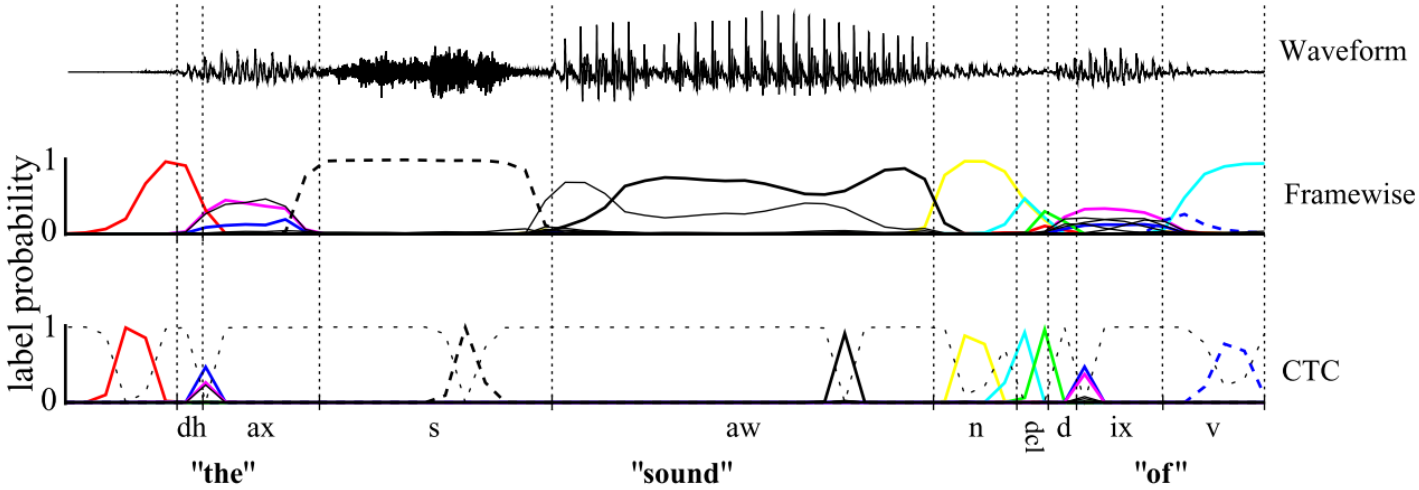
$$L = -\sum \log p(\mathbf{y}|\mathbf{x})$$

Graves, A., Fernández, S., Gomez, F., Schmidhuber, J., Connectionist temporal classification: labeling unsegmented sequence data with recurrent neural networks. 2006.

Connectionist Temporal Classification (CTC)

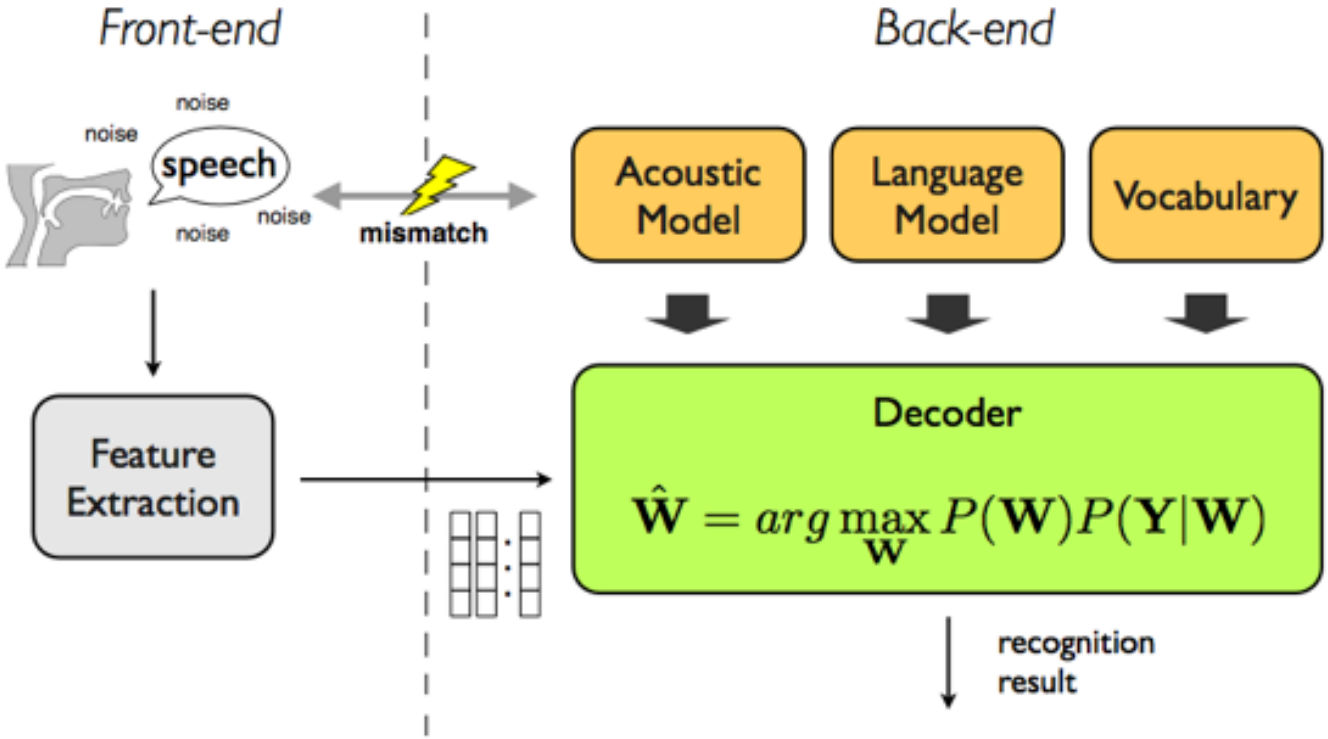


Predykcja CTC



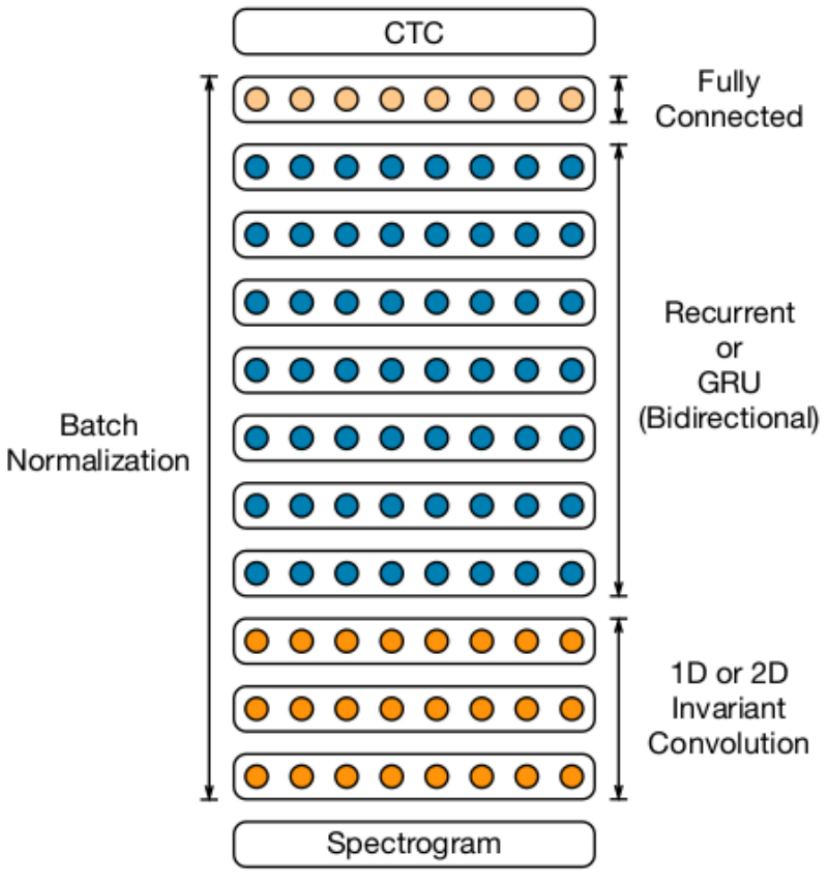
- trening z CTC generuje sekwencję pików etykiet (fonemów) przedzielonych etykietą 'blank'
- w treningu z aligmentem stosuje się funkcję kosztu CrossEntropy, przesunięcie granic fonemu względem poprawnego dopasowania powoduje błąd nawet gdy fonem jest poprawnie przewidziany
- wyjście CTC może być wykorzystane bezpośrednio (bez konieczności dalszego przetwarzania) do transkrypcji

Automatyczne rozpoznawanie mowy



Źródło grafiki: https://www.esat.kuleuven.be/psi/spraak/theses/08-09-en/MDT.php

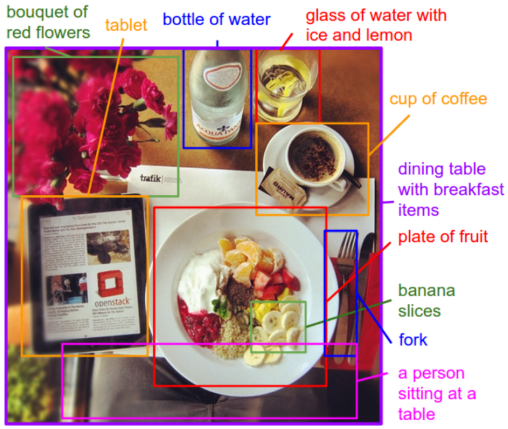
Przykład: Deep Speach 2

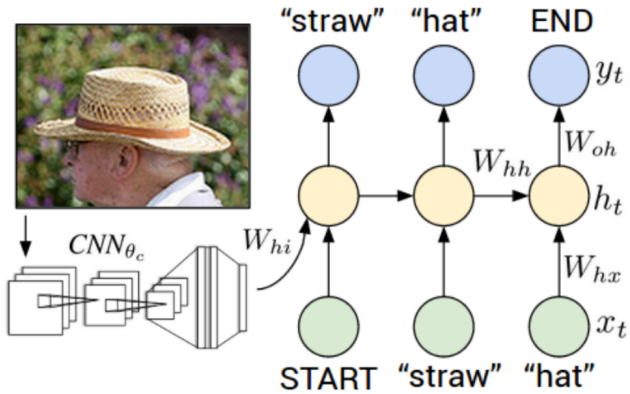


Read Speech			
Test set	DS1	DS2	Human
WSJ eval'92	4.94	3.60	5.03
WSJ eval'93	6.94	4.98	8.08
LibriSpeech test-clean	7.89	5.33	5.83
LibriSpeech test-other	21.74	13.25	12.69

Amodei, Dario, et al. "Deep speech 2: End-to-end speech recognition in english and mandarin."(2015).

NeuralTalk - generator opisów zdjęć





Multimodal Recurrent Neural Network ☞ NeuralTalk2

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