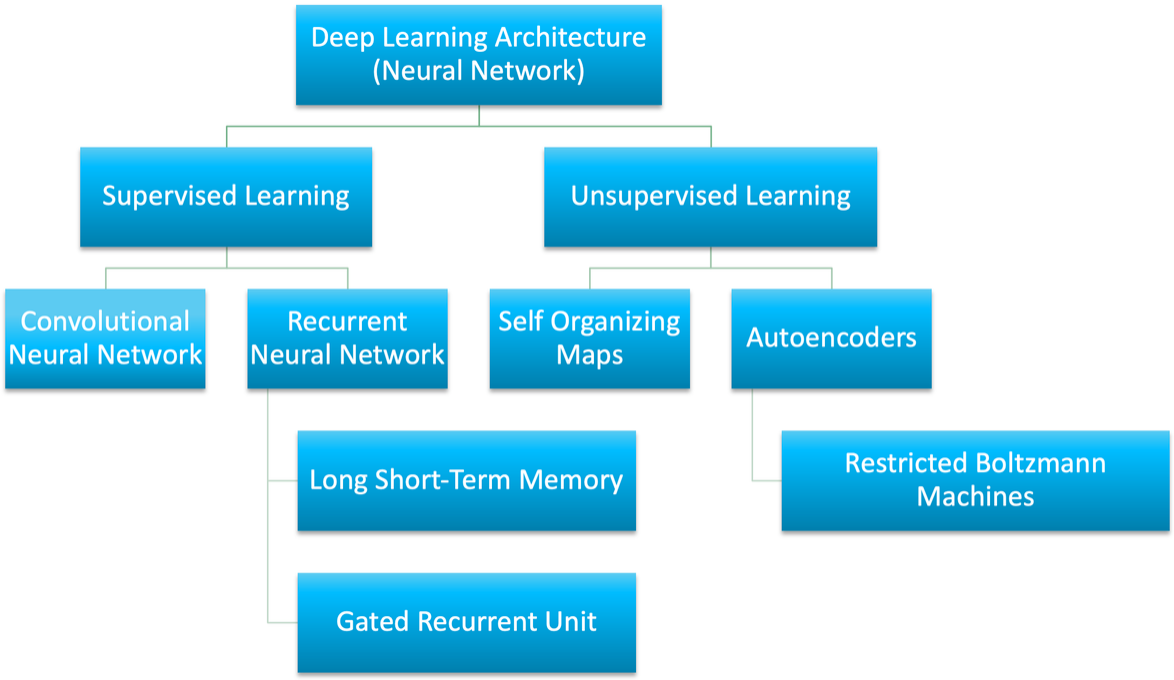
RNN&LSTM

أنواع خوارزميات التعلم العميق



Deep learning architectures - IBM Developer

الفرق بين CNN's و RNN's

خوارزميات CNN's(Convolutional Neural Networks)

- تقوم بتحليل spatial information أو visible patterns.
- تستخدم CNN's غالبا مع الصور.

خوارزميات RNN's) Recurrent Neural Networks)

- تقوم بتحليل temporal information أو sequential data بمعنى أننا لانحتاج لإدخال المعلومات الحالية فقط وإنما المعلومات السابقة أيضا، لذا نستخدم مايسمى Memory.
- تستخدم RNN's غالبا مع النصوص.
- أحيانا نستخدم كلا الخوارزميتين مثل Gesture Recognition حيث يتم استخدام CNN في مرحلة Feature Engineering ثم RNN عندما نحتاج Memory.

نماذج السلاسل الزمنية (Time Series Models)

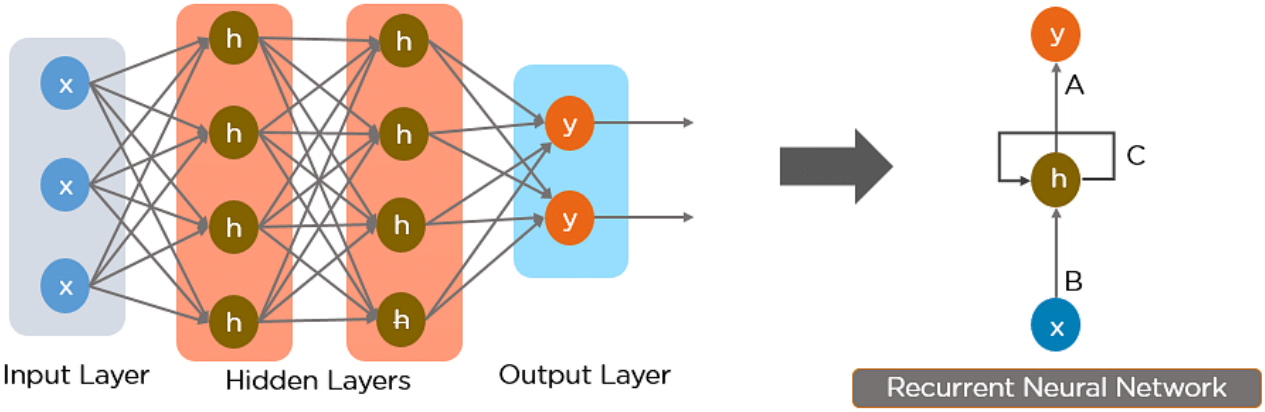
- تعود تطبيقات التعلم العميق إلى فترة 1940s، عندما تم نشر العالم Norbert Wiener كتاب Cybernetics.
- حاليا، التعلم العميق يعتبر في ذروته وذلك لعدة أسباب:
  - زيادة data sizes
  - زيادة model sizes
  - زيادة accuracy و complexity

هذا التطور في نماذج التعلم العميق أدى لظهور أساليب حديثة تساعد في نمذجة time series، وأبرز هذه الأمثلة:

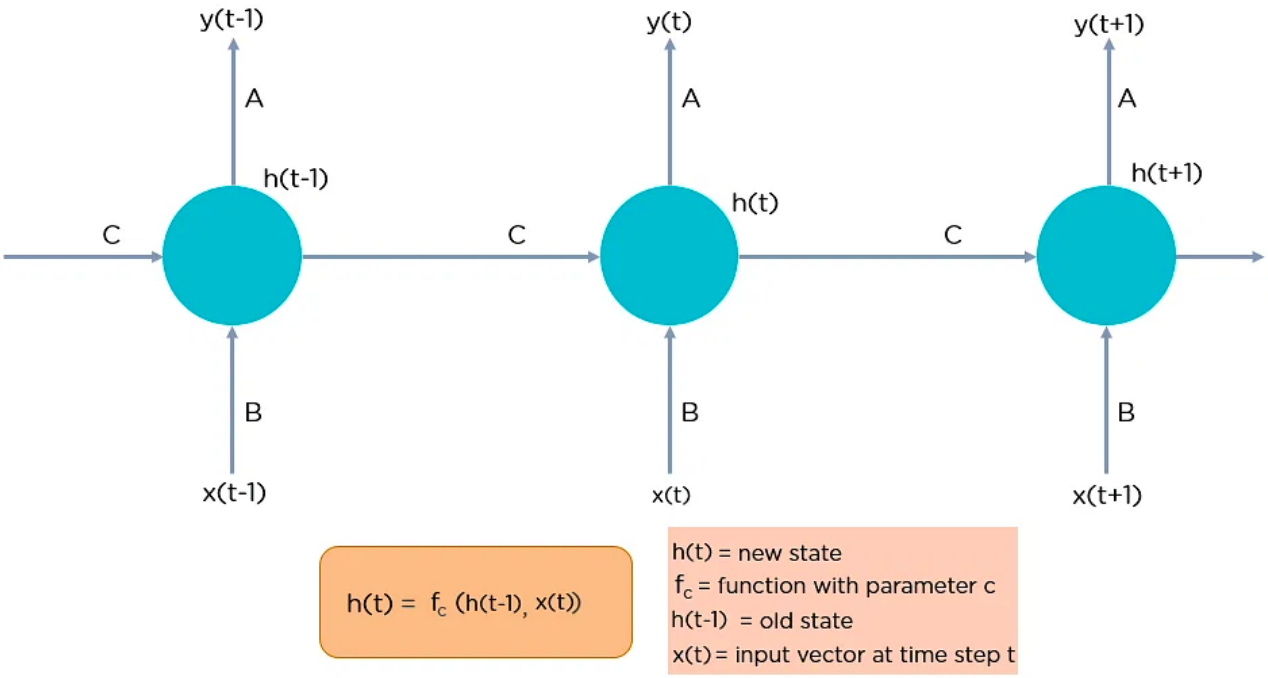
- خوارزمية RNN(recurrent neural network)
- خوارزمية LSTM(long short-term memory)

مايُميز هذه الخوارزميتين هو القدرة على احتواء longer time periods لعمل التنبؤ.

خوارزمية RNN (recurrent neural network)



Recurrent Neural Network (RNN) Tutorial: Types and Examples [Updated] | Simplilearn



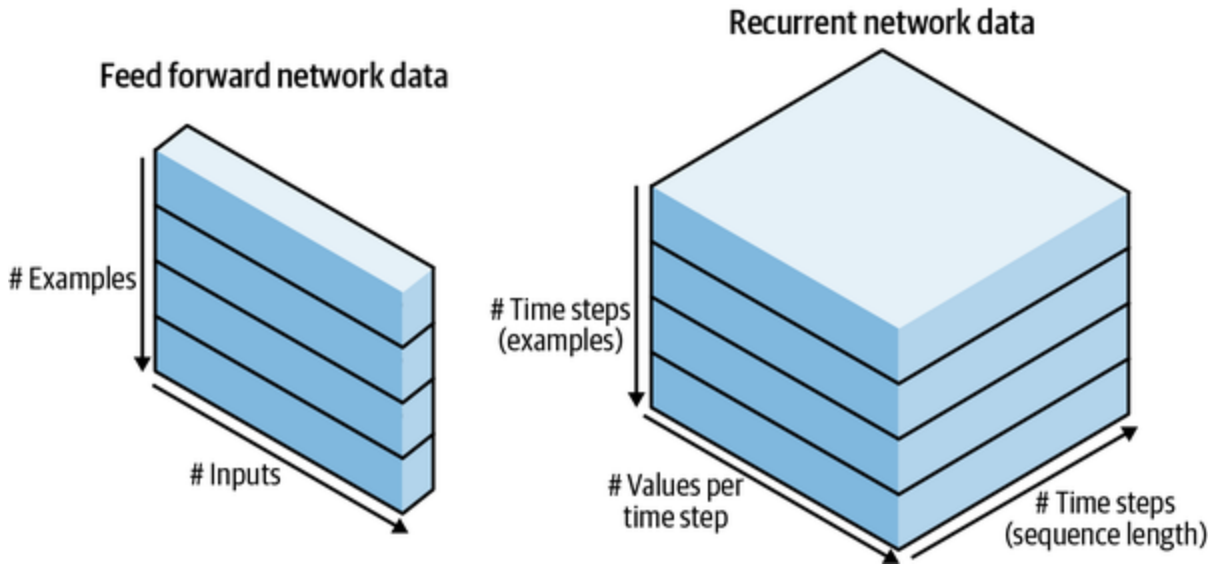
- هي خوارزمية بهيكلة تشابه neural network ولكن تحتوي one feedback connection على الأقل (تأتي على شكل loop) والهدف جعل الشبكة قادرة على تعلم السلاسل (sequences).
- وجود feedback connection يعني وجود خاصية مفيدة وهي الذاكرة (memory)، بمعنى أنه يمكن لـ RNN الاستفادة ليس فقط من بيانات الإدخال ولكن أيضًا من المخرجات السابقة للتنبؤ بالقيم التالية من السلسلة.

أشكال RNNs:

أولا: one-to-one RNN مكونة من single input و single output. تعتبر (most basic type).
ثانيا: One-to-many RNN مكونة من single input و multiple outputs.
ثالثا: Many-to-one RNN مكونة من multiple inputs و single output.
رابعا: Many-to-many RNN مكونة من multiple inputs and outputs. تعتبر (most complicated structure).

نبذة تاريخية عن RNN:

- قبل ظهور RNN، ظهرت feed-forward neural network ولكن مايُميز RNN هو قدرتها على التقاط temporal dependencies وذلك عن طريق تغذية neural network بوحدات hidden unit مرة أخرى مما يجعلها مناسبة لنمذجة بيانات time series.
- من الشكل التالي نلاحظ أن هيكلة RNN تحتوي time step بعكس feed-forward network.
- نلاحظ أيضا أن مدخلات RNN ثلاثية الأبعاد (three-dimensional input) مكونة من :
  - حجم Batch وهي تُمثل (number of observations/rows)
  - عدد Time steps وهي تُمثل (number of times to feed the model)
  - عدد features وهي تُمثل (number of columns of each sample)



RNN structure

دوال ACTIVATION FUNCTIONS:

- هي معادلات رياضية تستخدم لتحديد output في neural network structure.

أهم هذه الدوال:

- دالة Sigmoid

تدمج جزء بسيط من المخرجات أثناء عمل تغييرات في النموذج، قيمها بين 0 و 1.

$$\text{sigmoid}\,(x) = \frac{1}{1 + exp(-\sum_i w_i x_i - b)}$$

- دالة Softmax

تعتبر مشابهة لدالة sigmoid ويتم تطبيقها بشكل كبير في multi-classification problems.

$$\text{softmax}\left(x_i\right) = \frac{exp(x_i)}{\sum_i exp\left(x_i\right)}$$

- دالة Tanh

مناسبة للتعامل مع الأرقام السالبة، قيمها بين 1 و 1-.

$$\tanh\left(x\right) = \frac{sinh(x)}{cosh(x)}$$

- دالة Linear

تتيح بناء linear relationships بين المتغيرات independent و dependent، تناسب time-series models.

$$f(x) = wx$$

- دالة Rectified linear وتسمى (ReLu)

تكون قيمتها 0 إذا كان المُدخل 0 أو أقل من 0، أما إذا كان المُدخل أكبر من 0 ، فإنه يتم تمثيله بما يتماشى مع x.

$$\mathrm{ReLu(x)} = \max(0, x)$$

عيوب RNN drawbacks:

أولا: مشكلة vanishing

- أثناء تنفيذ backpropagation، يقل التدرج (gradient) عندما تزداد طبقات الشبكة، ومن المعروف أننا نقوم بتحديث weight بناء على قيمة gradient، إذا عندما تقل قيمة gradient أو تكون ثابته فهذا يعني ثبات قيم weight وبالتالي لايحدث تدريب للنموذج (يتعلم النموذج عن طريق تغيير التدرج) أي توقف optimization.

> ***The drawbacks of RNNs are well stated by Haviv et al. (2019):***
> *This is due to the dependency of the network on its past states, and through them on the entire input history. This ability comes with a cost—RNNs are known to be hard to train (Pascanu et al. 2013a).*

ثانيا: مشكلة exploding gradient

- على عكس المشكلة السابقة، تحدث هذه المشكلة أثناء عملية optimization، عندما تؤدي التعديلات البسيطة في backpropagation إلى تحديثات ضخمة في weight.

ثالثا: تدريب RNN يتطلب كمية كبيرة من البيانات.

رابعا: عدم قدرة RNN على معالجة very long sequences عندما تكون activation function هي tanh.

أمثلة لاستخدامات خوارزمية RNN:

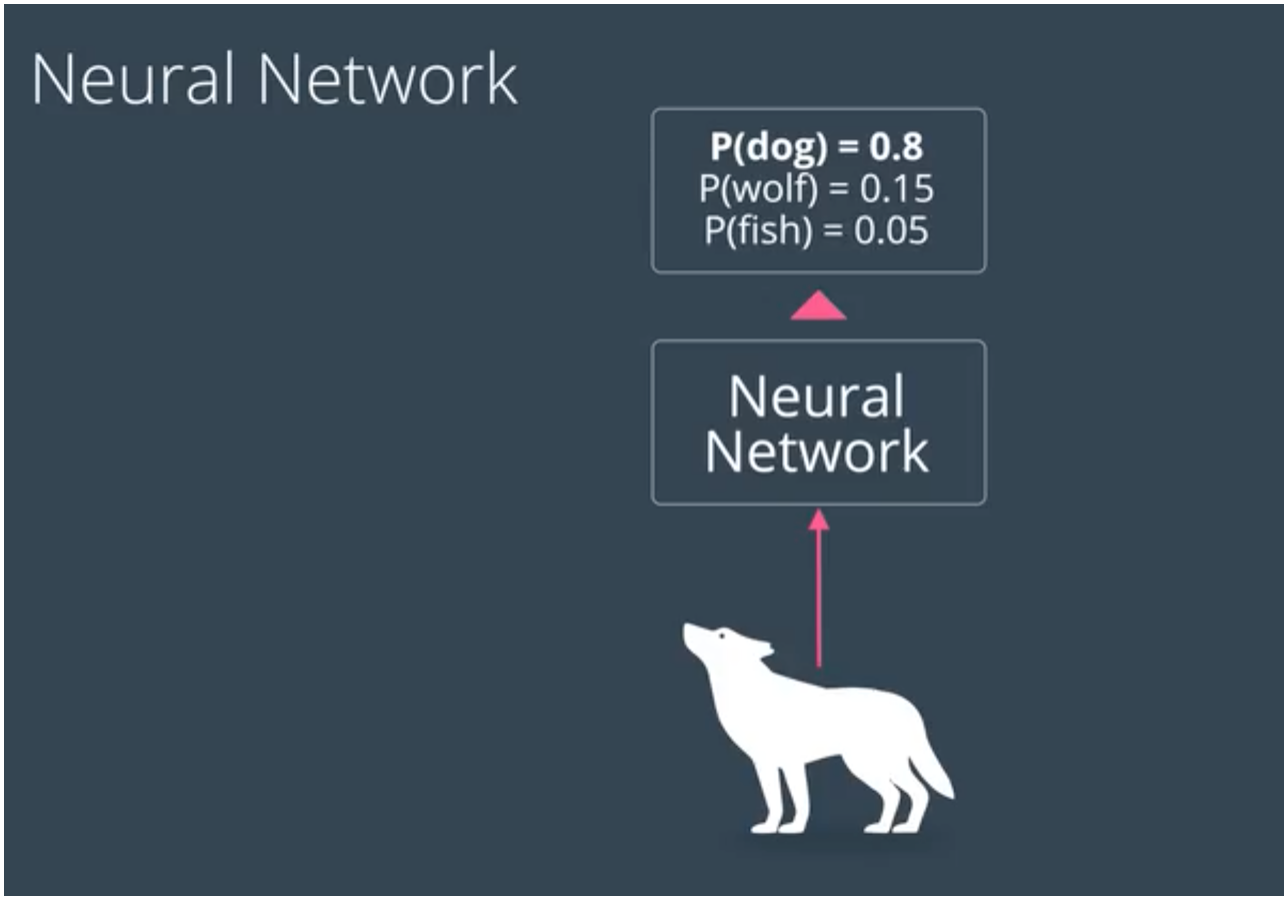
بالغالب يتم استخدامها في التطبيقات التي تعتمد على text processing أو text generation.

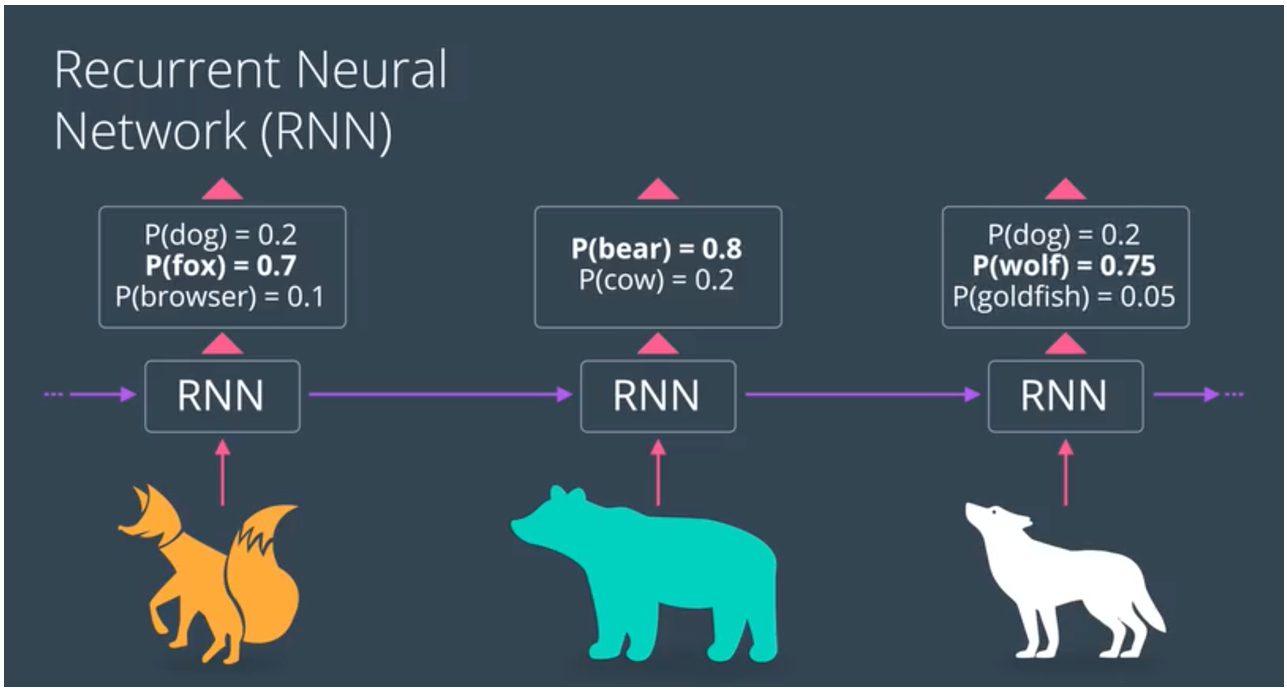
- تطبيقات Speech Recognition مثل: siri و alexa
- تطبيقات Machine Translation
- تطبيقات Question Answering و Chatbots
- تطبيقات Handwriting Recognition
- تطبيقات Time Series مثل: Prices Prediction و movie selection in Netflix

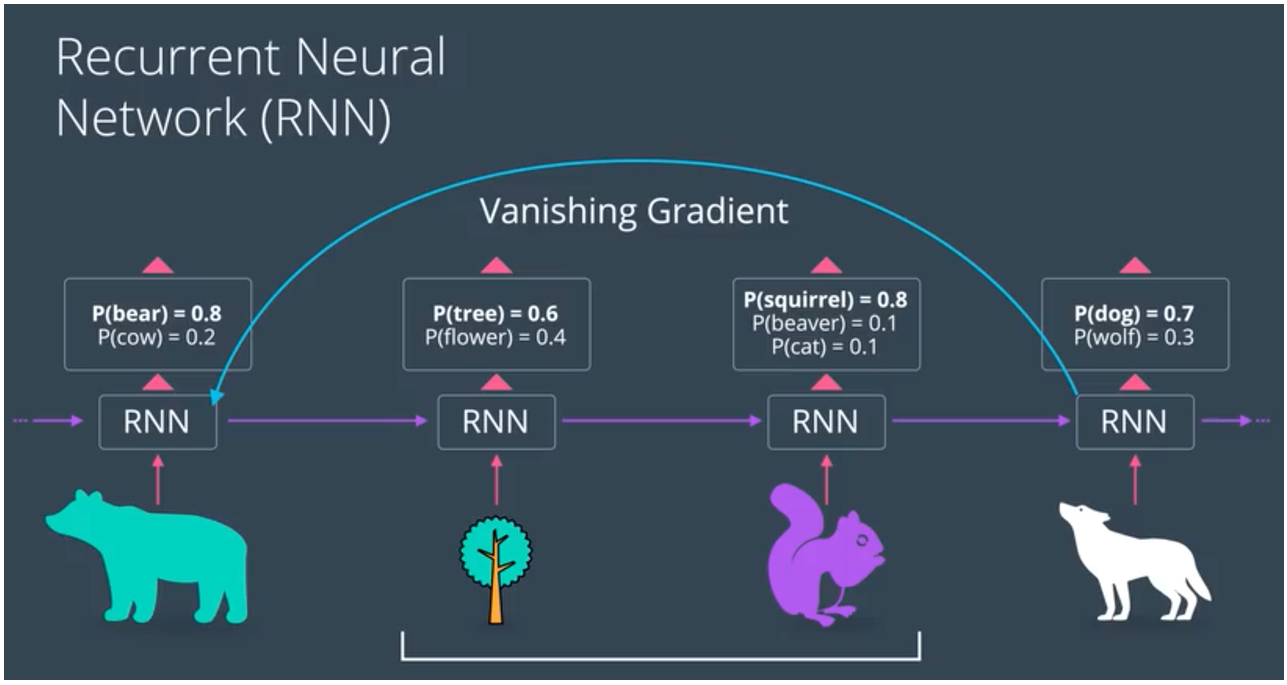
أمثلة أخرى:

- https://magenta.tensorflow.org/assets/sketch_rnn_demo/index.html
- https://www.youtube.com/watch?v=0FW99AQmMc8&t=1s
- https://arxiv.org/pdf/1511.06939.pdf
- https://openai.com/blog/dota-2/

(Long-Short Term Memory(LSTM خوارزمية





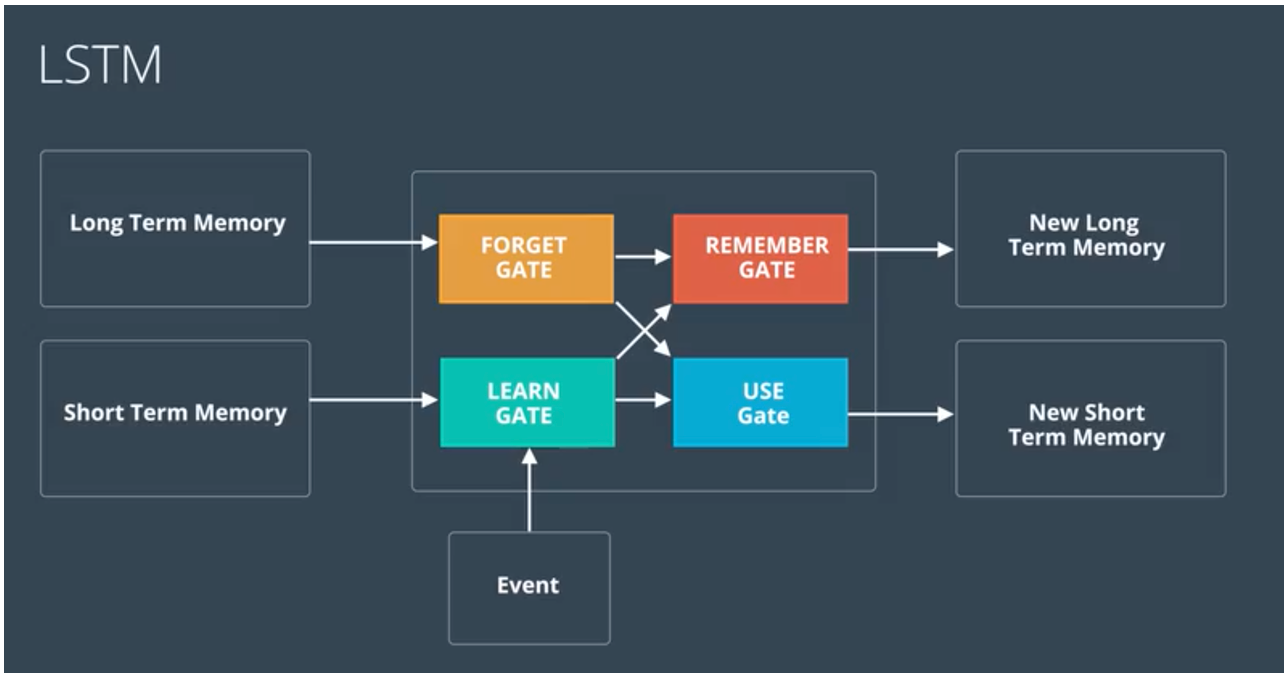


الخوارزمية	المدخلات	المخرجات
RNN's	Input Short-term Memory	Short-term Memory
LSTM's	Input Short-term Memory	Short-term Memory Long-term Memory

	Long-term Memory	

خوارزمية LSTM's

- هي إحدى خوارزميات التعلم العميق، تم تطويرها بواسطة1997) Hochreiter and Schmidhuber).
- تم اقتراحها كحل لمشكلة vanishing gradient problem.
- تعتمد بشكل أساسي على GRU) gated recurrent unit).
- تتكون GRU من two gates هي update و reset.
- عندما يكون لدينا highly important observation لانحتاج لعمل update على hidden state.
- أما في حال كان غير مهم سوف نقوم بعمل resetting the state.
- من أهم مميزات RNN قدرتها على ربط المعلومات السابقة والحالية، ولكن هذه الميزة تفشل في حالة Long-term dependencies والمقصود بذلك أن النموذج يتعلم من early observations.
- لذلك، خوارزمية LSTM تحاول حل هذه المشكلة عن طريق تجاهل المعلومات الغير ضرورية عن طريق مايسمى gates وبالتالي العمل بشكل أكثر كفاءة (more efficiently).



- هناك أربعة أنواع gates:
  - **أولا: Forget gates**

تأخذ المدخلات من Long-term Memory ثم تقوم بفرز المعلومات المفيدة وغير المفيدة باستخدام activation function وهي sigmoid بحيث تصبح قيمتها صفر إذا كانت المعلومات غير مفيدة.

  - **ثانيا: Learn gates**

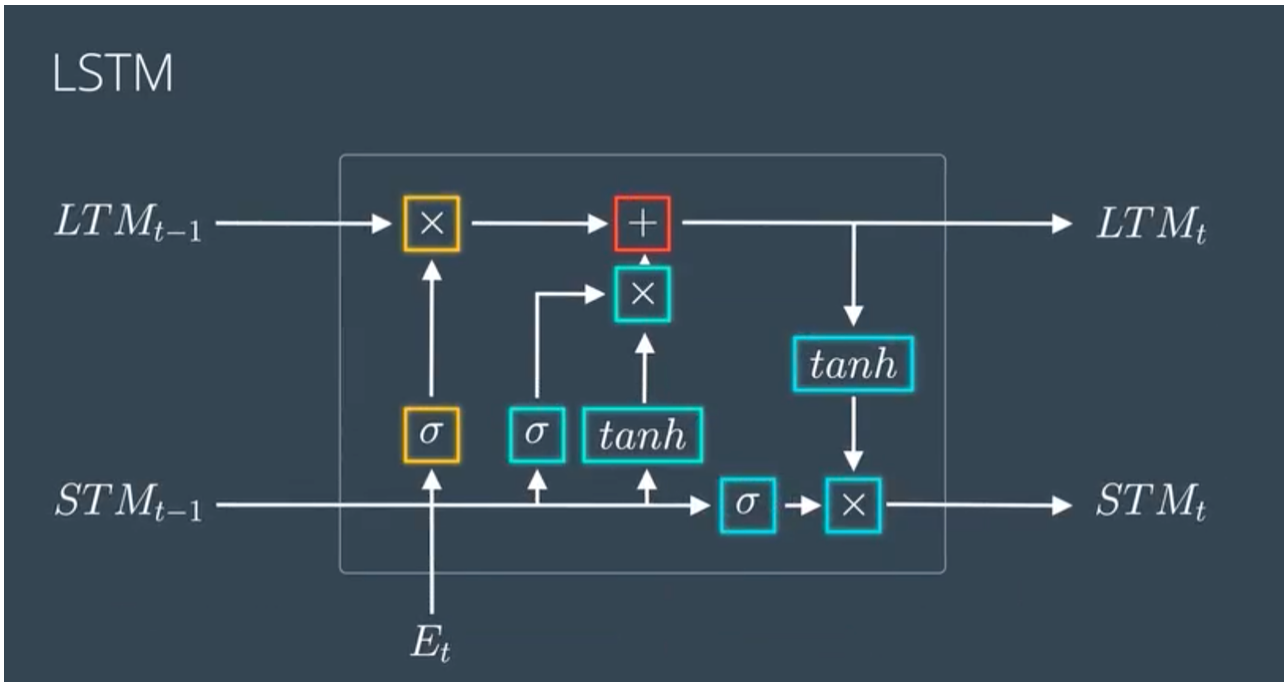
تأخذ المدخلات وهي Input و Short-term Memory بحيث تحتوي على المعلومات التي تم تعلمها مؤخرا ثم يتم تجاهل المعلومات غير المفيدة.

- **ثالثا: Remember gates**

تأخذ المدخلات وهي تمثل المعلومات المفيدة التي تم استخراجها من Learn and Forget gates ثم تستخدم هذه المعلومات لتحديث Long-term Memory.

- **رابعا: Use gates**

تأخذ المدخلات وهي تمثل المعلومات المفيدة التي تم استخراجها من Learn and Forget gates ثم تستخدم هذه المعلومات لعمل التنبؤ (output) ولتحديث Short-term Memory.



المعادلات بشكل مُفصل:

The Learn Gate

The output of the *Learn Gate* is $N_t i_t$ where:

$$N_t = \tanh(W_n[STM_{t-1}, E_t] + b_n)$$

$$i_t = \sigma(W_i[STM_{t-1}, E_t] + b_i)$$

Equation 1

The Forget Gate

The output of the *Forget Gate* is $LTM_{t-1} f_t$ where:

$$f_t = \sigma(W_f[STM_{t-1}, E_t] + b_f)$$

Equation 2

The Remember Gate

The output of the *Remember Gate* is:

$$LTM_{t-1} f_t + N_t \; i_t$$

Equation 3

(N_t, i_t and f_t are calculated in *equations 1* and 2)

The Use Gate

The output of the *Use Gate* is $U_t V_t$ where:

$$U_t = \tanh(W_u LTM_{t-1} f_t + b_u)$$

$$V_t = \sigma(W_v[STM_{t-1}, E_t] + b_v)$$

Equation 4

مصادر إضافية:

- http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- http://blog.echen.me/2017/05/30/exploring-lstms/
- https://github.com/abdullahkarasan/mlfrm
- https://learning.oreilly.com/library/view/machine-learning-for/9781492085249/ch03.html#idm45737245974704

- https://www.udacity.com/course/deep-learning-nanodegree--nd101