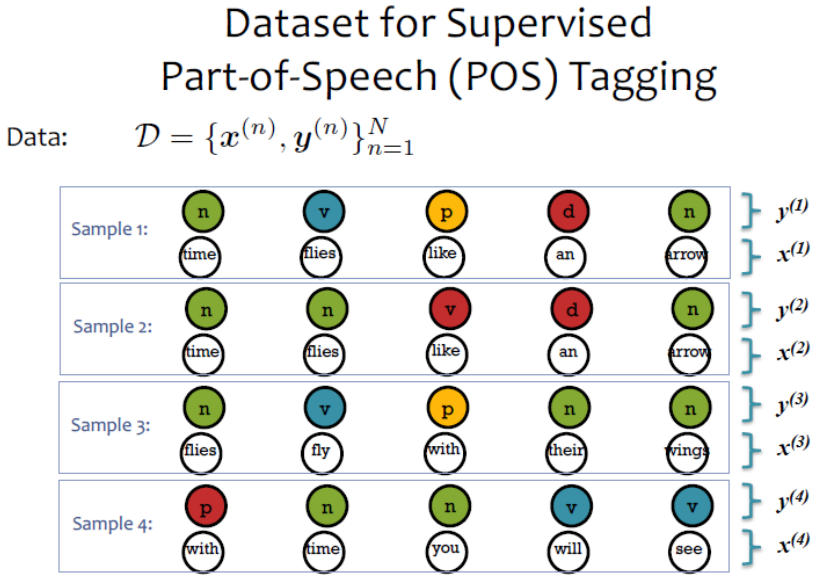
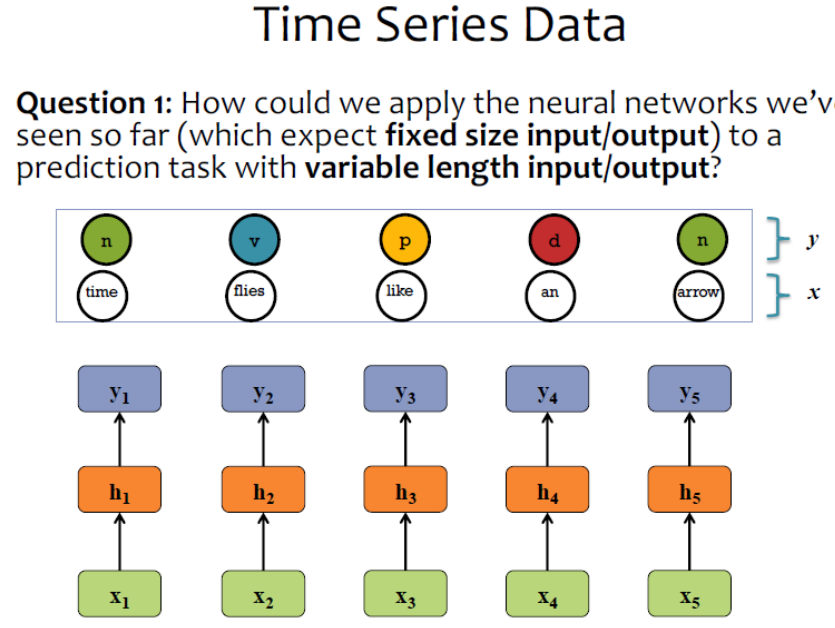
بسم الله الرحمن الرحيم

انهارده هنتكلم شويه عن الديب ليرننج و كمان هنفكر في ال PAC learning وده يعتبر أول theoretical justifications للي احنا بنعملو لحد دلوقت ... هنتكلم الاول علي high-level picture علي ال RNN لمدة نص ساعه .. والباقي هيخش في ال PAC .. يادوب .. احنا بدأنا نتكلم علي ال RNN وخدنا مثال علي ال time series dataset



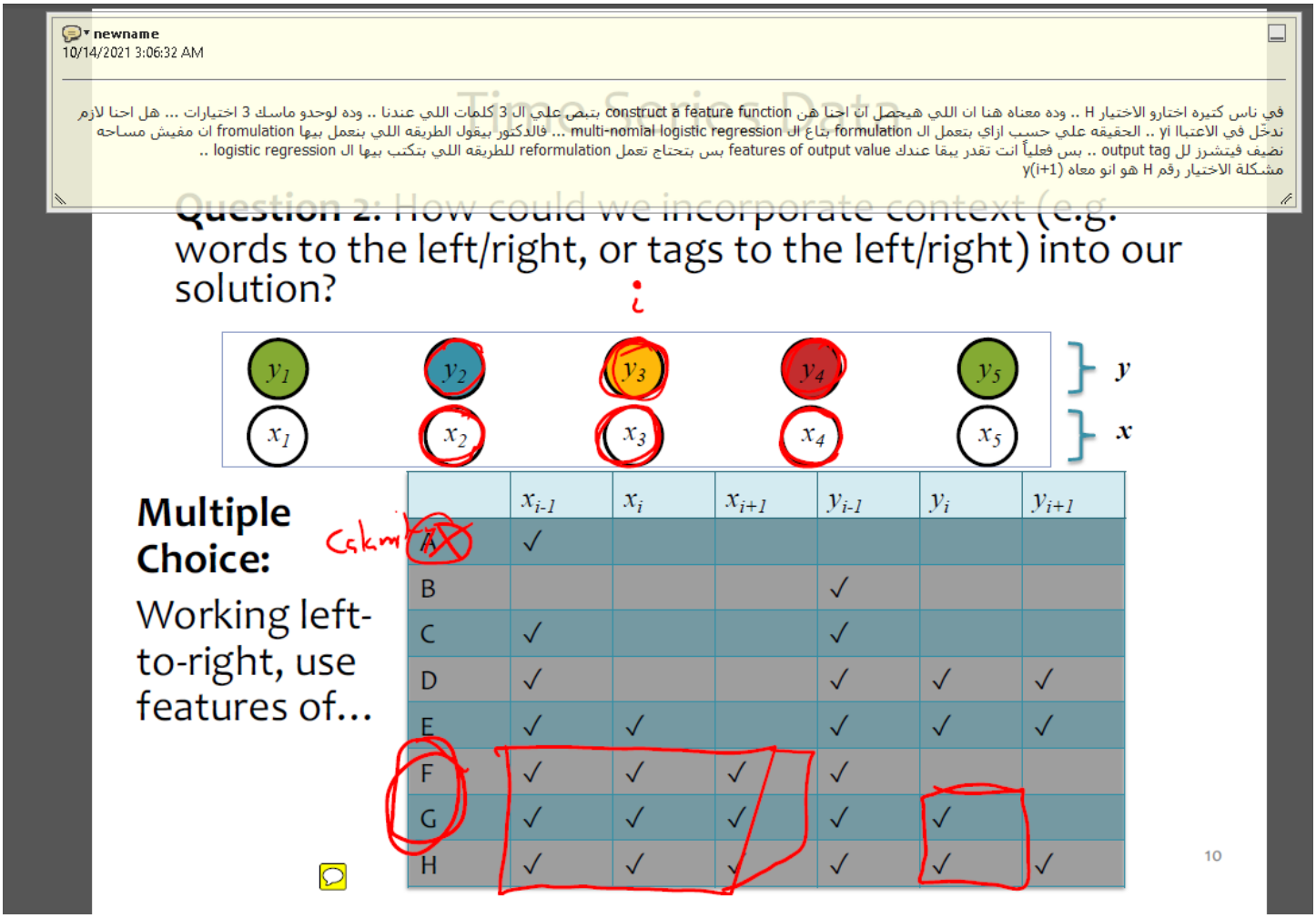
5

فانت ممكن يبقا عندك speech مثلا او hand writtern digits ... وهكذا .. وكان الدكتور قال ان اللي عاوزين نعملو هو اننا نفكر في ازاي نقدر conceptualize باستخدام ال fixed size neural network .. عشان نشتغل علي حاجه variable length input and output .. واللي عاوزين نفكر فيه هنا هو اننا نبني حاجه بتشتغل زي اللي كنا بنعملو في ال feature engineering discussion ... فهيبقا عندنا مثلا نويرال نتورك هتاخد كلمه مثلا و يبقا عندك hidden unit وت predict a preobability distribution over a part of speech وبعدين نكرر الحوار ده لكل واحد من ال speech tags اللي عندنا .. فهنلاقي عندنا 5 نويرال نتوركس منفصلين عن بعض .. في مشكله في الحوار ده ..



9

المشكله اللي الدكتور بيقولها هو انك لما هتيجي تبص علي كلمة flies .. ازاي هتعرف هي noun or verb؟ .. هل احنا بنتكلم علي الدبان .. ولا حاجه بتطير .. الطريقه اللي انت بتعرف بيها هي انهي واحده فيهم .. انك تبص علي كل كلمه في الجمله اللي عندك .. فانت عارف ان في الحاله ديه ان كلمه flies وراها كلمة time فهي فعلياً verb .. بس الصوره اللي فاتت من السلايد مش بتهاندل حاجه زي كدا .. انت كل اللي عارفه هو الكلمه اللي عندك دلوقت ... في poll question ... ازاي ن incorporate context ..



اقرا الكلام اللي مكتوب في السلايد . المهم الدكتور كان بيقول ان اخر اختيار هيبقا فيه مشكلة ال y(i+1) في الحاله ديه المشكله هتبقا ان مثلا يعني ازاي نبريدكت y3 احنا هنبص علي y2 and y4 .. في التست تايم انت معندكش أي تاج اصلا .. ال y2 and y4 هيجوا منين .. الدكتور بيقول ان اصلا predicted them .. فازاي عملت prediction for y4 انت استخدمت y3 and y5 ... فالدكتور قالك انت عشان تجيب y4 .. هتحتاج y3 .. وعشان تبرديكت y3 هتحتاج y4 .. في ال OS ده اسمو deadlock .. هنا اسمو circular dependency في الفيتشر فانكشن بتاعتنا .. بدل كدا اللي نقدر نعملو هو ان نعمل ال tagging problem عن طريق اننا نروح من الشمال لليمين .. نفس طريقة القرايه بتاعت الإنسان .. تعال نبص علي الاختيار بتاع F and G ... انت عمرك ما بتبص قدام هنا .. انت بس بتبص وراك ..

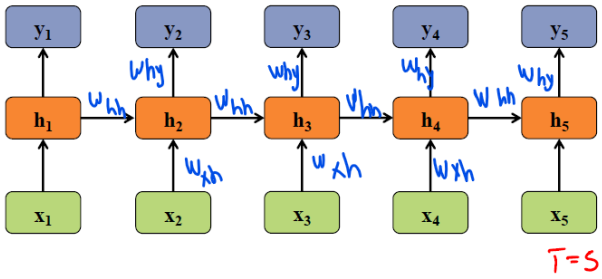
طيب ال RNN هتبقا نفس الفكره ديه .. هو انك ت incorporate contexts of x_is .. احنا هنفكر ازاي نحط all of the left and right context .. فتعال نشوف ازاي هنجيب كل الكونتكست اللي علي الشمال .. الفكره هنبني RNN اول y1 هتبقا زي standard feat forward NN عادي .. انما تاني y2 prediction هيبني ال hidden layer h2 عن طريق انو يبص علي h1 و x2 .. و هنفضل نتحرك بالطريقه ديه ... ازاي هنبني حاجه زي كدا وتبقا مفيده .. تعال بس نبص علي ال notation الاول ..

Recurrent Neural Networks (RNNs)

inputs: $\mathbf{x} = (x_1, x_2, \ldots, x_T), x_i \in \mathcal{R}^I$
hidden units: $\mathbf{h} = (h_1, h_2, \ldots, h_T), h_i \in \mathcal{R}^J$
outputs: $\mathbf{y} = (y_1, y_2, \ldots, y_T), y_i \in \mathcal{R}^K$
nonlinearity: $\mathcal{H}$

Definition of the RNN:

$$h_t = \mathcal{H}\left(W_{xh}x_t + W_{hh}h_{t-1} + b_h\right)$$
$$y_t = W_{hy}h_t + b_y$$

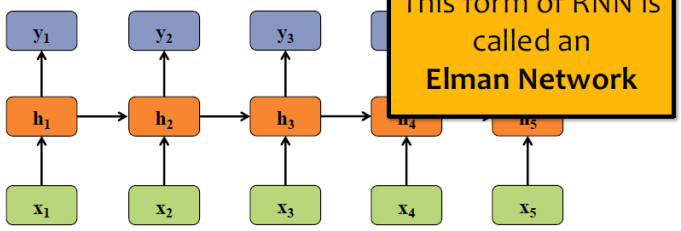


Recurrent Neural Networks (RNNs)

inputs: $\mathbf{x} = (x_1, x_2, \ldots, x_T), x_i \in \mathcal{R}^I$
hidden units: $\mathbf{h} = (h_1, h_2, \ldots, h_T), h_i \in \mathcal{R}^J$
outputs: $\mathbf{y} = (y_1, y_2, \ldots, y_T), y_i \in \mathcal{R}^K$
nonlinearity: $\mathcal{H}$

Definition of the RNN:

$$h_t = \mathcal{H}\left(W_{xh}x_t + W_{hh}h_{t-1} + b_h\right)$$
$$y_t = W_{hy}h_t + b_y$$

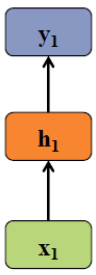


Recurrent Neural Networks (RNNs)

inputs: $\mathbf{x} = (x_1, x_2, \ldots, x_T), x_i \in \mathcal{R}^I$
hidden units: $\mathbf{h} = (h_1, h_2, \ldots, h_T), h_i \in \mathcal{R}^J$
outputs: $\mathbf{y} = (y_1, y_2, \ldots, y_T), y_i \in \mathcal{R}^K$
nonlinearity: $\mathcal{H}$

Definition of the RNN:

$$h_t = \mathcal{H}\left(W_{xh}x_t + W_{hh}h_{t-1} + b_h\right)$$
$$y_t = W_{hy}h_t + b_y$$



- If $T=1$, then we have a standard feed-forward **neural net with one hidden layer**
- All of the deep nets from last lecture required **fixed size inputs/outputs**

Background

A Recipe for Machine Learning

1. Given training data:

$$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$$

2. Choose each of these:
   - Decision function

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

   - Loss function

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

3. Define goal:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

4. Train with SGD:
(take small steps opposite the gradient)

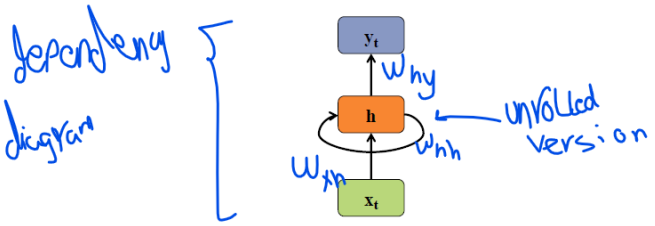
$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta_t \nabla \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

Recurrent Neural Networks (RNNs)

inputs: $\mathbf{x} = (x_1, x_2, \ldots, x_T), x_i \in \mathcal{R}^I$
hidden units: $\mathbf{h} = (h_1, h_2, \ldots, h_T), h_i \in \mathcal{R}^J$
outputs: $\mathbf{y} = (y_1, y_2, \ldots, y_T), y_i \in \mathcal{R}^K$
nonlinearity: $\mathcal{H}$

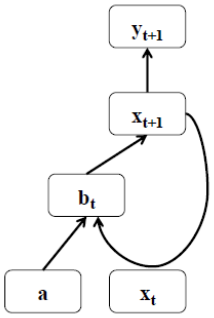
Definition of the RNN:

$$h_t = \mathcal{H}\left(W_{xh}x_t + W_{hh}h_{t-1} + b_h\right)$$
$$y_t = W_{hy}h_t + b_y$$



Background: Backprop through time

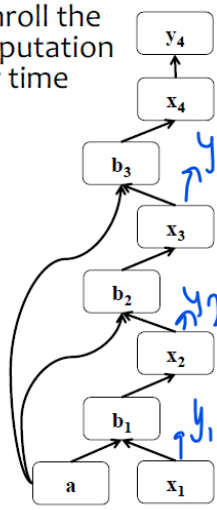
Recurrent neural network:



BPTT:

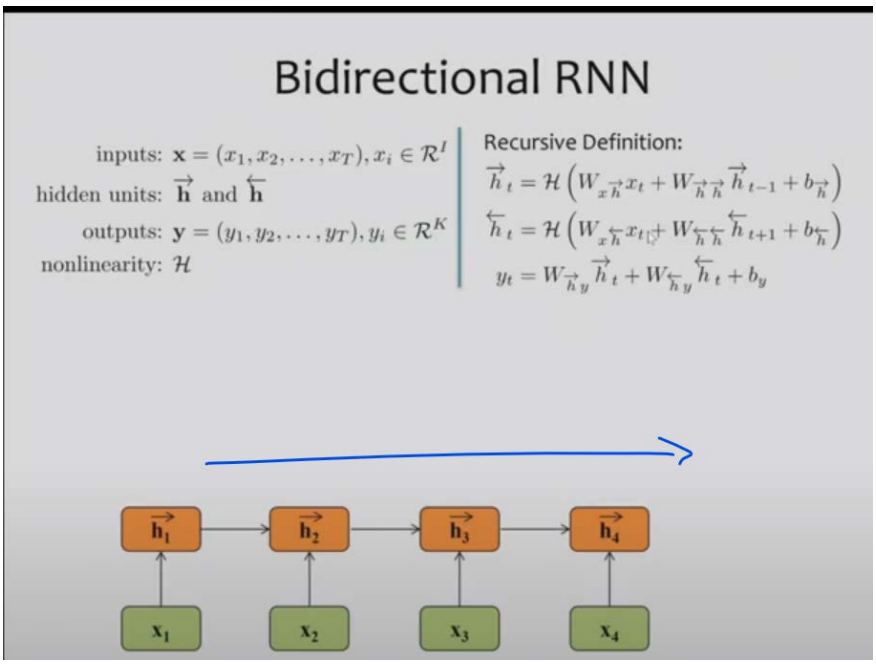
1. Unroll the computation over time

(Robinson & Fallside, 1987)
(Werbos, 1988)
(Mozer, 1995)



2. Run backprop through the resulting feed-forward network

السلايد الي فاتت ديه هي ال computational graph ...



Bidirectional RNN

inputs: $\mathbf{x} = (x_1, x_2, \ldots, x_T), x_i \in \mathcal{R}^I$

hidden units: $\overrightarrow{\mathbf{h}}$ and $\overleftarrow{\mathbf{h}}$

outputs: $\mathbf{y} = (y_1, y_2, \ldots, y_T), y_i \in \mathcal{R}^K$

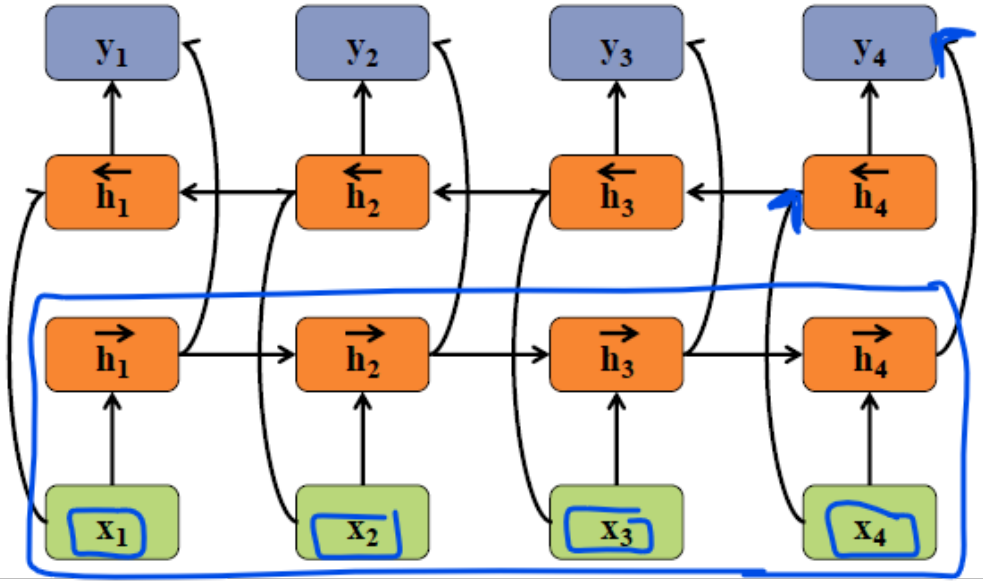
nonlinearity: $\mathcal{H}$

Recursive Definition:

$$\overrightarrow{h}_t = \mathcal{H}\left(W_{x\overrightarrow{h}} x_t + W_{\overrightarrow{h}\overrightarrow{h}} \overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}\right)$$

$$\overleftarrow{h}_t = \mathcal{H}\left(W_{x\overleftarrow{h}} x_t + W_{\overleftarrow{h}\overleftarrow{h}} \overleftarrow{h}_{t+1} + b_{\overleftarrow{h}}\right)$$

$$y_t = W_{\overrightarrow{h}y} \overrightarrow{h}_t + W_{\overleftarrow{h}y} \overleftarrow{h}_t + b_y$$



كدا وانت رايح لقدام انت h3 هو هيددين ريبرزنتيشن لحد x3 .. وانت راجع بقا h3 شمال ده ريبرزنتيشن لكل اللي علي اليمين فانت لما بت predict y3 .. بتاخد الاتنين h3 left and right arrow وتجيب ال distribution over output class y3 .. طيب لو عوزنا نعمل deep RNNs ... عندنا layers كتيره فوق بعض ..

Deep RNNs

inputs: $\mathbf{x} = (x_1, x_2, \ldots, x_T), x_i \in \mathcal{R}^I$

outputs: $\mathbf{y} = (y_1, y_2, \ldots, y_T), y_i \in \mathcal{R}^K$

nonlinearity: $\mathcal{H}$

Recursive Definition:

$$h_t^n = \mathcal{H}\left(W_{h^{n-1}h^n} h_t^{n-1} + W_{h^n h^n} h_{t-1}^n + b_h^n\right)$$

$$y_t = W_{h^N y} h_t^N + b_y$$

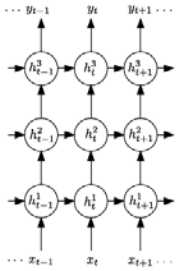


Figure from (Graves et al., 2013)

23

Deep Bidirectional RNNs

inputs: $\mathbf{x} = (x_1, x_2, \ldots, x_T), x_i \in \mathcal{R}^I$

outputs: $\mathbf{y} = (y_1, y_2, \ldots, y_T), y_i \in \mathcal{R}^K$

nonlinearity: $\mathcal{H}$

- Notice that the upper level hidden units have input from **two previous layers** (i.e. wider input)
- Likewise for the output layer
- What analogy can we draw to DNNs, DBNs, DBMs?

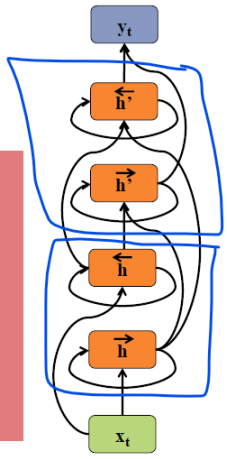


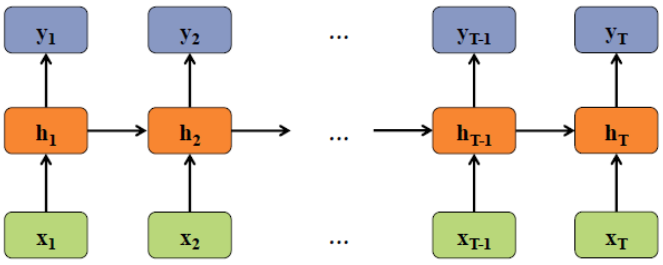
Figure from (Graves et al., 2013)

24

Long Short-Term Memory (LSTM)

Motivation: problem in Elman

- Standard RNNs have trouble learning long distance dependencies
- LSTMs combat this issue



ال LSTMs بيعالج ال long distance dependencies ... افترض ان عندك second-to-last tag وانت محتاج تبقا فاهم اول كلمتين فعلا بشكل كويس .. غير كدا انت مش هتفهم الإنتربتيشن ... فكر في "time flies like an arrow" ... لو انت فهمت غلط الverb flies فده هي break بقيت الفهم بتاع الجمله .. في الانجليزي تقدر تجيب dependencies that stretch the string .. يعني مثلا

"time flies, as you know it always has, like an arrow"

فالمشكله ديه لو جيت تبص ع السلايد .. هتلاقي ان مشكلة ال vanishing gradient problem عندك .. بص علي ال influence بتاع اول كلمه .. وبص علي الكلمه اللي قبل الأخيره .. هتلاقي ان ال standard RNN بتوضح في الصوره ان ال influence بيقع قدام في النتورك بتاعتك .. معناه ان لو انت فهمت ال tag اللي قبل الأخير غلط .. يبقا انت محتاج تعدل ال weight بتاع أول كلمه .. وده هيبقا صعب انك تعملو back-prop ..

Long Short-Term Memory (LSTM)

Motivation:

- Vanishing gradient problem for Standard RNNs
- Figure shows sensitivity (darker = more sensitive) to the input at time t=1

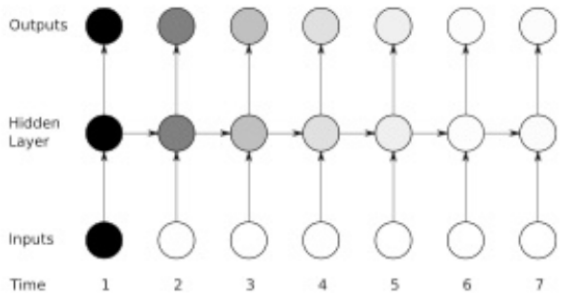


Figure from (Graves, 2012)

Long Short-Term Memory (LSTM)

Motivation:

- LSTM units have a rich internal structure
- The various "gates" determine the propagation of information and can choose to "remember" or "forget" information

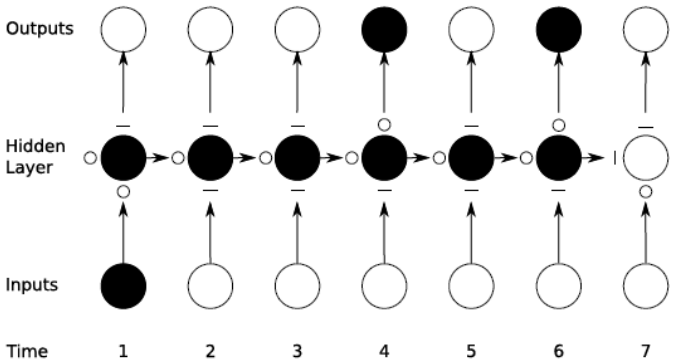


Figure from (Graves, 2012)

Long Short-Term Memory (LSTM)

هنا بنستخدم لوبة gates

Motivation:

- LSTM units have a rich internal structure
- The various "gates" determine the propagation of information and can choose to "remember" or "forget" information

0 → open gate

→ closed gate

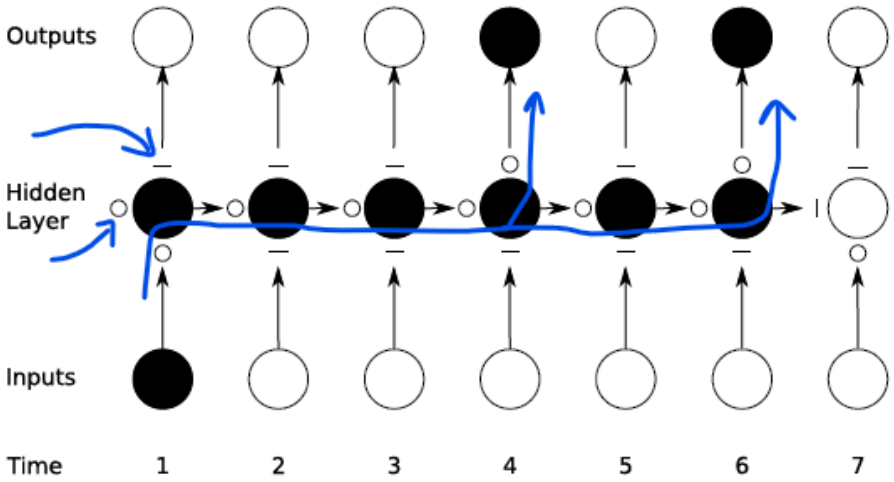
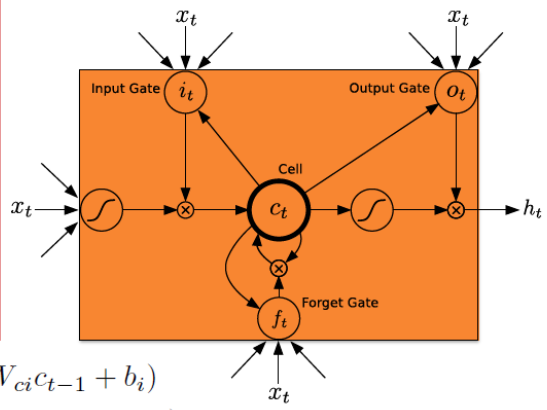


Figure from (Graves, 2012)

Long Short-Term Memory (LSTM)

- **Input gate:** masks out the standard RNN inputs
- **Forget gate:** masks out the previous cell
- **Cell:** stores the input/forget mixture
- **Output gate:** masks out the values of the next hidden



$$i_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i\right)$$
$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right)$$
$$c_t = f_t c_{t-1} + i_t \tanh\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right)$$
$$o_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o\right)$$
$$h_t = o_t \tanh(c_t)$$

Figure from (Graves et al., 2013)

طيب دلوقت هنتكلم علي ال learning theory .. الهدف من ده نفهم هو ليه اللي قلناه قبل كدا في المحاضرات فعلا بيشتغل ... .. "انا بحب الدكتور ده جداً .. بيفكرني بديفيد ميلان بتاع CS50" ... طيب لو عندك كلاسيفير ليه صفر ترينج ايرور .. ايه هو اللي تقدر تقولو علي الجينيريلايزيشن ايرور او الترو

ايرور ... او سؤال تاني .. لو عندك كلاسيفير ليه ترينج ايرور قليل ... ايه اللي يتقال في الترو ايرور .. منين ما بتتكلم علي ترو ايرور وتست ايرور .. انت دايماً مهتم بالاوفر فيتنج ... وواحده من الحاجات اللي انت بتعملها هو ال regularization المهم الدكتور بيقول هل احنا نقدر نجيب حاجه ك theoretical ... justification for our ways of combating overfitting

Questions For Today

1. Given a classifier with **zero training error**, what can we say about **true error** (aka. generalization error)?
   (Sample Complexity, Realizable Case)
2. Given a classifier with **low training error**, what can we say about **true error** (aka. generalization error)?
   (Sample Complexity, Agnostic Case)
3. Is there a **theoretical justification for regularization** to avoid overfitting?
   (Structural Risk Minimization)

39

تعال نجاوب علي اول سؤال .. عندك داتا سورس .. بيديك unlabeled examples وده بيخش ل expert ... وبعدين يقوم موديه لل learning algorithm .. وبعدين ده يتعلم decision boundary ... تعال نبص علي نوعين الإيرورز الي عندنا ..

Two Types of Error

1. True Error (aka. **expected risk**)

$$R(h) = P_{\mathbf{x} \sim p^*(\mathbf{x})}(c^*(\mathbf{x}) \neq h(\mathbf{x}))$$

Prob

This quantity is always **unknown**

2. Train Error (aka. **empirical risk**)

$$\hat{R}(h) = P_{\mathbf{x} \sim \mathcal{S}}(c^*(\mathbf{x}) \neq h(\mathbf{x}))$$

$$= \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}(c^*(\mathbf{x}^{(i)}) \neq h(\mathbf{x}^{(i)}))$$

$$= \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}(y^{(i)} \neq h(\mathbf{x}^{(i)}))$$

We can **measure** this on the training data

where $\mathcal{S} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)})\}_{i=1}^{N}$ is the training data set, and $\mathbf{x} \sim \mathcal{S}$ denotes that $\mathbf{x}$ is sampled from the empirical distribution.

42

الترو ايرور هنا هو الدكتور قال انو بنحسب ال probability of making a mistake ل example randomly sample from p star و ال p star هنا هي ال unknown distribution علي الإكسات اللي عندك ... فالترينج ايرور والتست ايروور هم زي بعض .. الفرق بس بيبقا where the actual distribution over Xs comes from ... هل هي ال unknown p star و لا هي ال training sample S ... لو رجعنا لل PAC/SLT model ...

PAC / SLT Model

We've also referred to this as the "Function Approximation View"

1. Generate instances from *unknown* distribution p^*

$$\mathbf{x}^{(i)} \sim p^*(\mathbf{x}), \forall i \quad (1)$$

2. Oracle labels each instance with *unknown* function c^*

$$y^{(i)} = c^*(\mathbf{x}^{(i)}), \forall i \quad (2)$$

3. Learning algorithm chooses hypothesis $h \in \mathcal{H}$ with low(est) training error, $\hat{R}(h)$

$$\hat{h} = \underset{h}{\operatorname{argmin}}\, \hat{R}(h) \quad (3)$$

4. Goal: Choose an h with low generalization error $R(h)$

Three Hypotheses of Interest

1

The **true function** c^* is the one we are trying to learn and that labeled the training data:

2

$$y^{(i)} = c^*(\mathbf{x}^{(i)}), \forall i \quad (1)$$

The **expected risk minimizer** has lowest true error:

$$h^* = \underset{h \in \mathcal{H}}{\operatorname{argmin}}\, R(h)$$

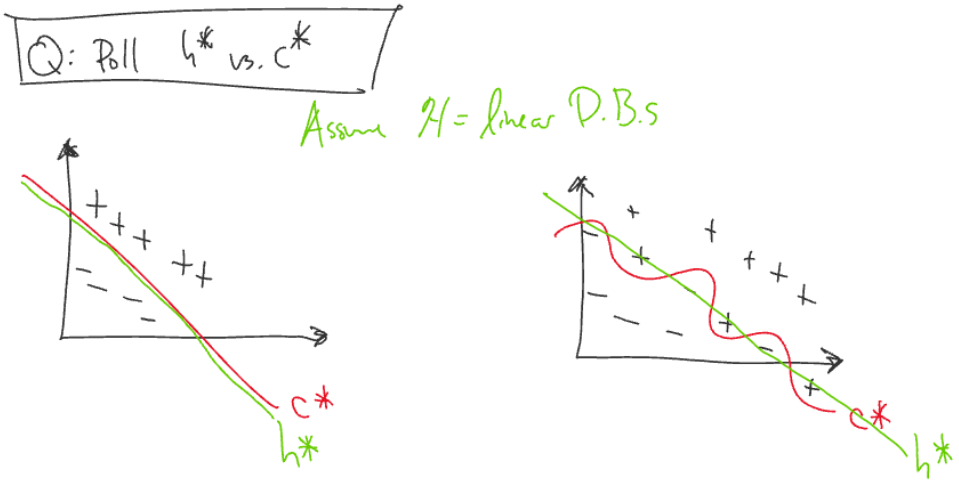
min True error

Question:
True or False:
h* and c* are always equal.

2

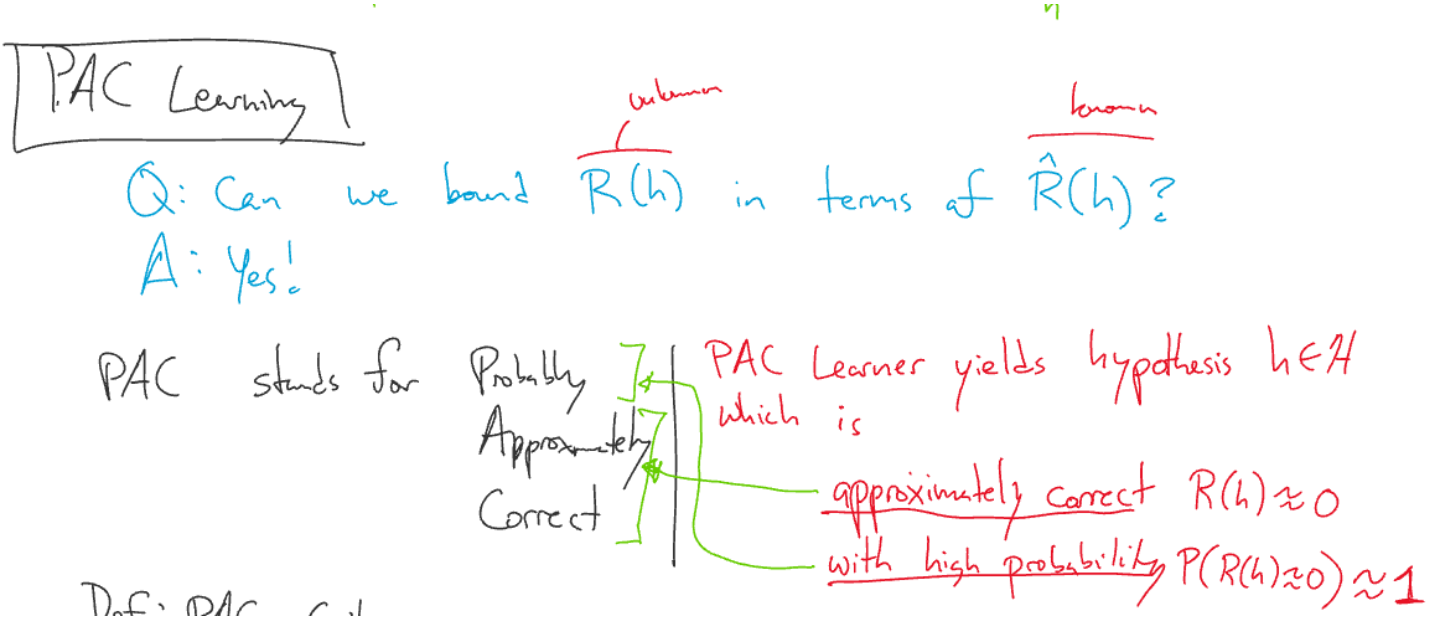
The **empirical risk minimizer** has lowest training error:

$$\hat{h} = \underset{h \in \mathcal{H}}{\operatorname{argmin}}\, \hat{R}(h) \quad (3)$$

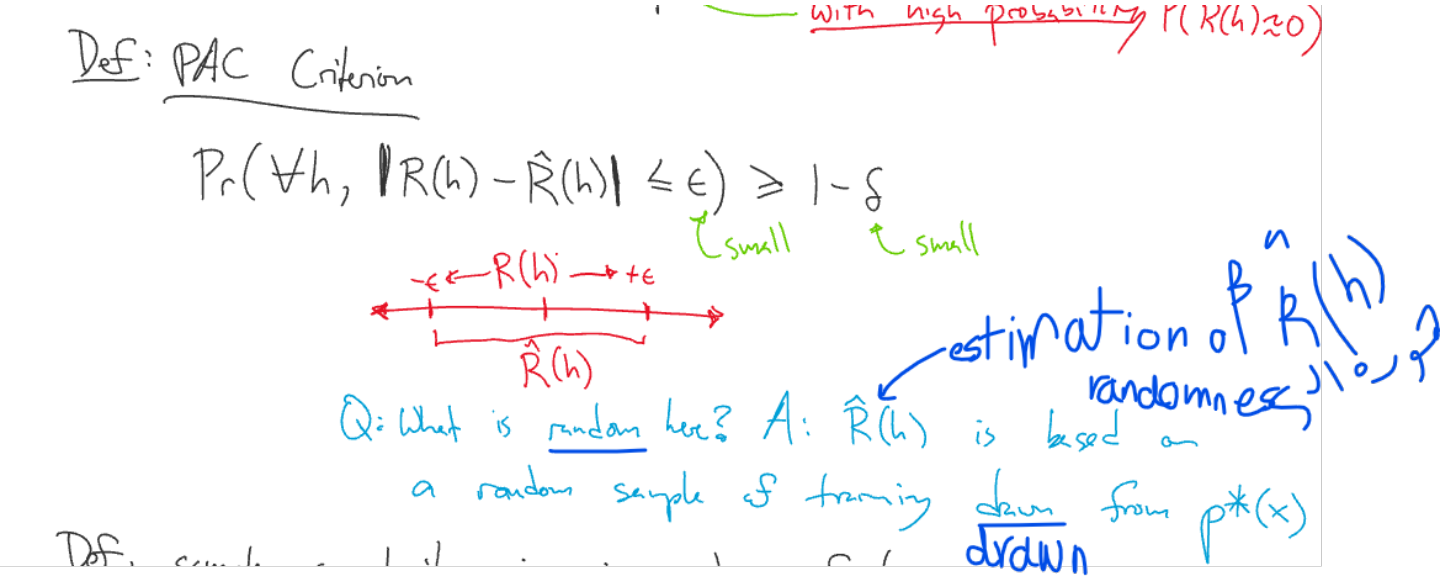


الدكتور رجع للسلايد بتاعت الايرور وقال انت متقدرش تحسب ال h star عشان انت اصلا متعرفش ايه هي ال Rh .. فال Pac learning هو الهدف منو انو نجاوب علي سؤال هيخلينا نلف وندور في الكلام شويه .. Can we boundR(h) in terms of R hat(h)... الاجابه اه ... R(h) ديه انت مش عارفها

انما ال R hat(h) ده عارفينو .. فاحنا هدفنا ن bound unknown quantitiy in terms of known quantity .. ال PAC هنا اختصار ل probably apprixmately correct .. وال PAC learner .. هو هيبقا hypothesis h from set of Hypothesis H .. وديه بتبقا approximately correct باحتماليه عاليه ...

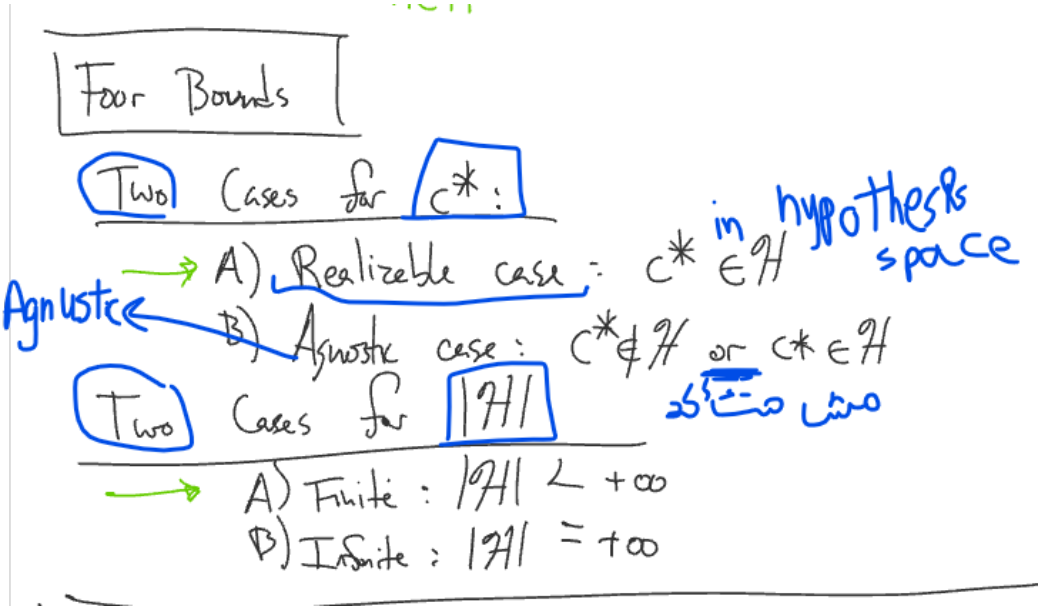


ايه المقصود من approximately correct .. اللي احنا مهتمين بيه هو ال true error فبالنسبه لل true error ايه هو معني ان نبقي approximately correct .. معناه ان R(h) تقريبا بصفر ... ايه المقصود ب high probability .. هنا P(R(h)) close to 1 ... طيب هنتكلم علي تعريف ال PAC criterion وهو ان probability of all hypotheses having true error minus training error ك absolute value اقل من او بتساوي إبسيلون .. و ده بيبقا اكبر من او بيساوي واحد ماينص دلتا .. الدلتا و الإبسيلون ارقام صغيره مثلا 0.01 ..... اللي بنقولو ان عشان نبقا تقريبا صح .. لو التريننج ايرور كان صغير اللي هو هنا ال R hat of h .. احنا مش عاوزين ال true error يبقا بعيد عنو انم يبقا قريب منو بمقدار ابسيلون .. فال R(h) هي fixed unknown true error ... وعندك رينج من سالب ابسيلون ل موجب ابسيلون .... التريننج ايرور هيبقا في الرينج ده ... احنا ليه دخلنا في سكة ال proability .. هي الاحتماليه هنا بتبقا ف ايفنت .. ايه الايفنت اللي هنا .... فين ال randomness اللي في الموضوع ...



Def: sample complexity is min number of train examples N st. the PAC Criterion is satisfied for ϵ and δ

Def: a hypothesis is consistant w/the training data if $\hat{R}(h) = 0$ $h \in H$ D



أول نظريه:

Thm.1: Sample Complexity (Realizable, Finite |H|)

$$N \geq \frac{1}{\epsilon}\left[\ln(|H|) + \ln(1/\delta)\right]$$ labeled examples are sufficient to ensure that w/prob. $(1-\delta)$ all $h \in H$ with $\hat{R}(h) = 0$ have $R(h) \leq \epsilon$

هنا عندنا finite H .. فاحنا نفكر ده زي ايه بالظبط .. حاجه كدا زي الديسيجن تريز ... فانت طلع عدد كتير من التريز لحد ديبث معين يعني .. أول نظرية بتقول لو عندنا N كان اكبر من اللوج بتاع حجم ال hypotheses space و N ده هو عدد ال labeled examples .. وكان الأمثله ديه كانت كفيله انها ت ensure ان باحتماليه عاليه حوالي واحد ماينص دلتا .. كل ال hypothesis in H ليهم صفر ترينج ايرور اللي هو R hat = 0 ليهم true error R(h) اقل من او بيساوي epsilon .. الجمله ديه قويه .. بتقول ان لو احنا عدد كافي من الأمثله اللي ليها labels .. فبالتالي باحتماليه 1 – delta كل ال hypotheses اللي ليهم صفر ترينج ايرور برضو هيبقا ليهم low true error .. ... ازاي نجيب الاثبات بتاع theorem 1 ...

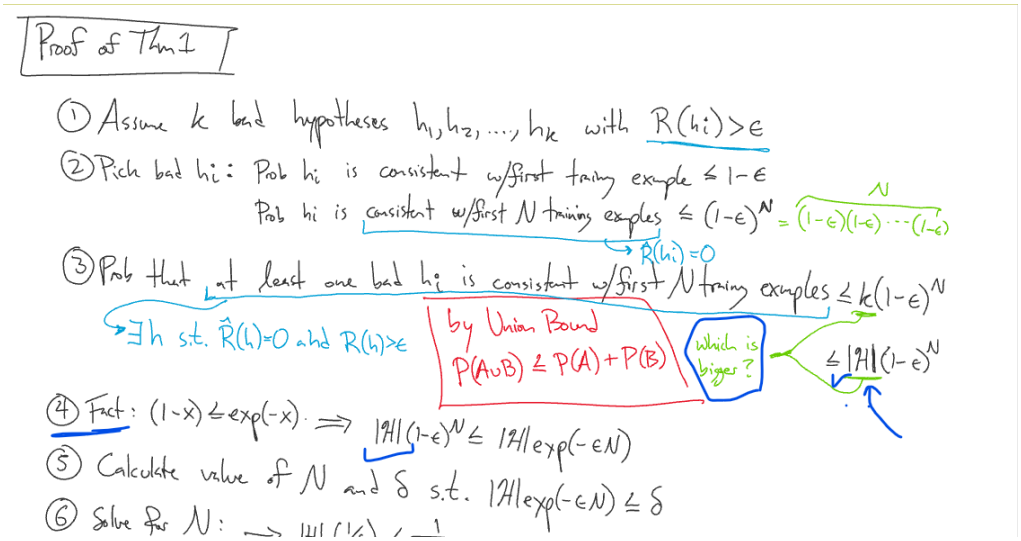
تعال نفترض عندنا hk ..... h1, h2, K bad hypotheses .. معني انو وحش ان التروو ايرور اكبر من ال epsilon .. افترض انك اخترت one specific bad hi ... انا عاوز اعرف ايه احتمالية ان hi هتبقا consistent مع أول training example .. ايه هو الرقم ده .. ده معناه يعني انها تبقا

صح في الترينج اكزامبل اللي عندك دلوقت يعني .. ايه هي احتمالية انك تعمل إيرور في أول مثال عندك بالكلاسيفير السيء ده ...؟ ايه هي ال R(hi) ... true error = probability of making a mistake on a randomly chosen example

فبالتالي ايه هي الاحتماليه ان a bad h(i) will make a mistake on a randomly chosen example? .. ؟ .. هتبقا أكبر من إبسيلون .. ديه احتماليه انك تعمل غلطة ... طيب ايه هي الاحتماليه انك تبقا consistent مع أول مثال عندك يعني انك فعلا عرفت انو صح .. هتبقا اقل من او بيساوي واحد ماينص ابسيلون

طيب ايه احتمالية ان ال hi هتبقا consistent مع أول N training examples ... هتبقا اقل من او بيساوي واحد ماينص ابسيلون الكل مرفوع لل N ...

طيب ايه احتمالية ان علي الاقل واحده من ال bad his تبقا consistent مع اول N training examples .. الدكتور راح يتكلم علي ال union bound اللي هو P(AUB) هي اقل من P(A) + P(B) .. وقال ان A هو ان عندك first of the bad hypotheses be consistent with the data .. B هو 2^{nd} bad hypothesis is consistent with the data .. فهنا انت بتقول ان احتمالية ان واحد علي الاقل يبقا وحش .. هي ال union .. فهنا انت هتجمع عدد k ..



... and δ s.t. $|H|\exp(-\epsilon N) \leq \delta$

⑥ Solve for N:

$\Rightarrow |H|\left(\frac{1}{\delta}\right) \leq \frac{1}{\exp(-\epsilon N)}$

$\Rightarrow \ln(|H|) + \ln\left(\frac{1}{\delta}\right) \leq \ln(1) - \ln(\exp(-\epsilon N))$

$\Rightarrow \frac{1}{\epsilon}\left[\ln(|H|) + \ln\left(\frac{1}{\delta}\right)\right] \leq N$ $\quad \epsilon N$

Statement 1

Assume Stat. 1, then:

with probability w/prob δ $\exists h$ s.t. $R(h) > \epsilon$ and $\hat{R}(h) = 0$ "bad"

Contrapositive

$A \rightarrow B$

$\neg B \rightarrow \neg A$

w/prob $(1-\delta)$

all $h \in H$ with $R(h) > \epsilon$ (A) have $\hat{R}(h) > 0$ (B) $\neg$bad

$\Rightarrow$ all $h \in H$ with $\hat{R}(h) = 0$ ($\neg$B) have $R(h) \leq \epsilon$ ($\neg$A) not bad

if learner returns $h \in H$ w/zero training error then w/prob $(1-\delta)$ h has $\leq \epsilon$ true error