التعلم العميق Deep Learning

- **الدرس الأول : التعلم العميق و الشبكات العصبية**
  - الأسبوع الأول : مقدمة للتعلم العميق
  - الأسبوع الثاني : أساسيات الشبكات العصبية
  - الأسبوع الثالث : الشبكات العصبية المجوفة
  - الاسبوع الرابع : الشبكات العصبية العميقة
- **الدرس الثاني : تطوير الشبكات العميقة : المعاملات العليا**
  - الأسبوع الأول : السمات العملية للتعلم العميق
  - الأسبوع الثاني : الحصول علي القيم المثالية
  - الأسبوع الثالث : ضبط قيم الشبكات العميقة
- **الدرس الثالث : هيكلية مشاريع الـ ML**
  - الأسبوع الأول : استراتيجيات الـ ML - 1
  - الأسبوع الثاني : استراتيجيات الـ ML - 2
- **الدرس الرابع : الشبكات العصبية الملتفة CNN**
  - الأسبوع الأول : أساسيات الشبكات العصبية الملتفة
  - الأسبوع الثاني : حالات عملية من الشبكات العصبية الملتفة
  - الأسبوع الثالث : التعرف علي الأشياء
  - الاسبوع الرابع : التعرف علي الوجه
- **الدرس الخامس : الشبكات العصبية المتكررة RNN**
  - الأسبوع الأول : مفهوم الشبكات العصبية المتكررة
  - الأسبوع الثاني : المعالجة اللغوية الطبيعية NLP
  - الأسبوع الثالث : نماذج التتابع

درس 5: الشبكات العصبية المتكررة RNN

الأسبوع الثالث : نماذج التتابع Sequence Models

==========

نماذج التتابع يمكن أن تتم بر استخدام آليات الانتباه , وهو ما سيساعد الخوارزم علي فهم اين يقوم بتركيز انتباهه بناء علي معطيات متتابعة .

كما سنتعرف علي آلية التعرف علي الحديث speech recognition و كيفية التعامل مع البيانات الصوتية

-

نتعرف أولا علي ما يسمي نموذج التتابع الي التتابع sequence to sequence model , وهو الذي يستخدم بشكل كبير في عدد من التطبيقات مثل التعرف علي الاصوات و ترجمتها الي نصوص .

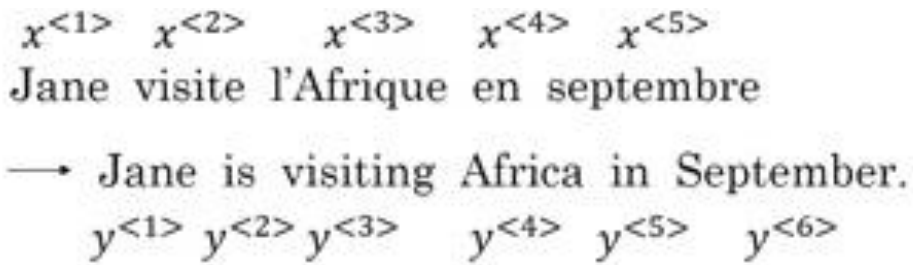
كمثال , لو كان لدينا جملة بالفرنسية :

Jane visite l'Afrique en septembre

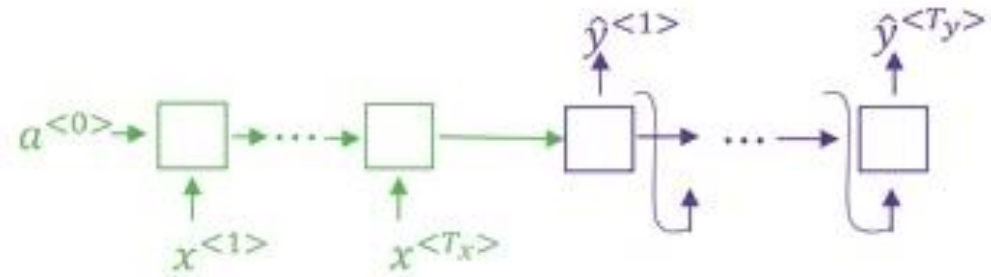
ونريد أن نترجمها إلي الإنجليزية , والتي ستكون :

Jane is visiting Africa in September.

فأول خطوة , تكون بجعل كل كلمة من كلمات الفرنسي وهي المدخلات تسمي $x^{<1>} \ldots$, وكذلك كل كلمة من كلمات المخرجات ستكون $y^{<1>} \ldots$ هكذا :

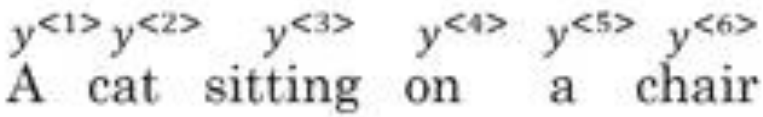


بعدها يمكن بناء شبكة متكررة RNN بحيث تتناول المدخلات معا في الجزء الأخضر , و تسمي : المشفر encoder ثم يتم تحويلها الي الجزء الثاني من الشبكة الذي سيقوم بالتفسير decoder وهو الجزء الازرق الذي سيخرج كلمات انجليزية .

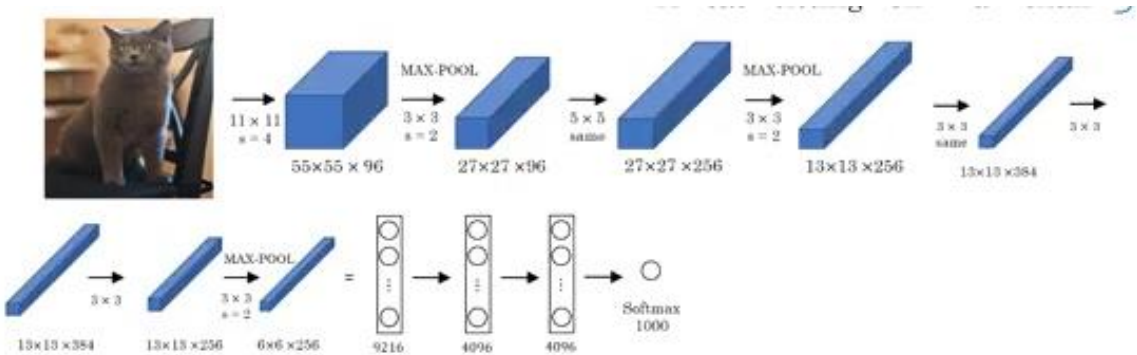


مثال آخر هو ما يسمي : توصيف الصور image captioning , وهو الذي يتناول الصور كمدخلات , وتكون مهمته ان يكتب وصف مناسب لها حسب المحتوي , فمثلا هذه الصورة :

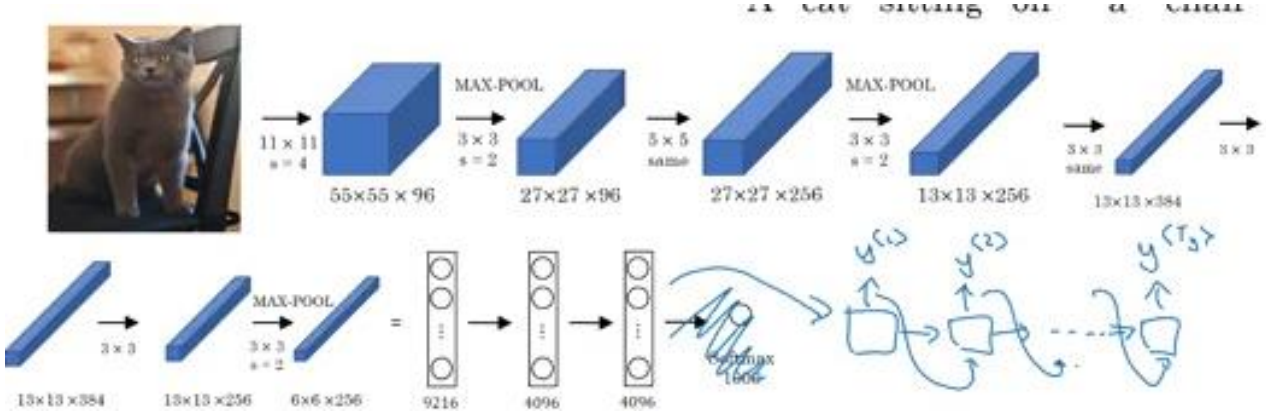
سيكون النص المناسب لها هو :



ويبدأ الأمر بإدخال صورة القطة في أحد نماذج الشبكات الملتفة CNN و ليكن نموذج alexnet , والذي كان يتناول صورة القط و وينتهي في سوفتماكس يسحدد هل الصورة قط ام لا



لكننل سنقوم بإلغاء هذا السوفت ماكس , و التعامل مع كل هذه الشبكة علي أنها encoder , ثم إيصالها بشبكة RNN تعتبر decoder و التي ستقوم بإخراج كلمات متتالية تعتبر الوصف الخاص بالصورة :

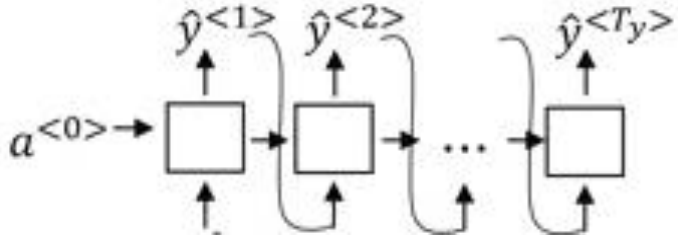


وهذه الفكرة خطيرة جدا , خاصة اذا ما تم ربطها مع تكنيك word sentiment و هو الخاص بتقييم مشاعر الكلمات , فعبر وضع كاميرا معينة تلتقط صورة , ثم تتحول الي وصف , ثم الي مشاعر , يمكن تحديد كيفية التعامل مع الشئ المقابل .

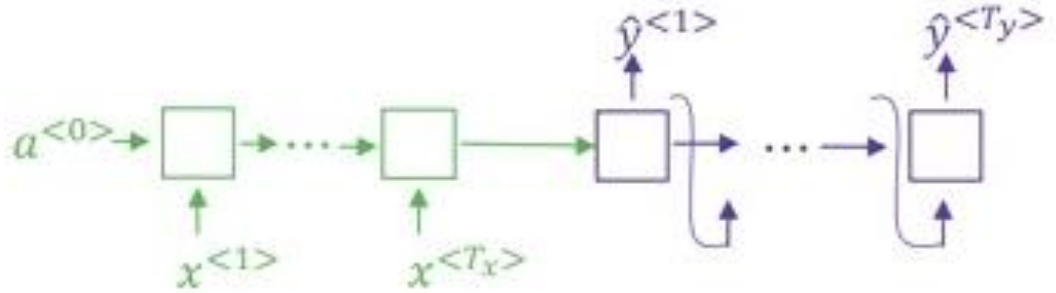
-

وعلي الرغم من أن نموذج التتابع للتتابع sequence to sequence هو له نفس الهيكل والاساس للشبكات المتكررة , والتي قمنا باستخدامها في التطبيقات اللغوية , إلا أن هناك عدد من الإختلافات . .

و الفرق الاساسي , ان النموذج اللغوي language model (والذي استخدمناه في كتابة شعر او قصة او اخبار) لا يأخذ أي مدخلات , او قد يأخذ مدخل عشوائي , ويكون اول مدخل له a<0> هي اصفار ليس اكثر



بينما التتابع للتتابع , فهو يحتوي أولا علي شبكة متكررة (الخضراء) التي تقوم بعمل مشفر decoder ثم يتم الحاقها النموذج السابق , والذي سيكون بدلا من إدخال أصفار في a<0> سيتم إدخال المخرج من شبكة المشفر الخضراء



فالنموذج الأول يقوم بإخراج جمل عشوائية , بينما الثاني يقوم بإخراج جمل تعتمد علي جمل المدخل .

لذا فإن قياس إحتمالية المخارج في النموذج الأول تكون عشوائية إلي حد ما و تكتب هكذا :

P(y<1> , y<2> , y<3> , y<4> . . .)

حيث قيم y هي الكلمات الناتجة .

بينما اقياس احتمالية نموذج التتابع للتتابع , سيكون :

P(y<1> , y<2> , y<3> . . . | x<1>, x<2>, x<3>, . . .)

اي احتمالية كتابة كلمات كذا كذا , بمعلومية أن المدخل هي كلمات كذا كذا

فلو أن الجملة الفرنسية المطلوب ترجمتها هي :

Jane visite l'Afrique en septembre

فقد تكون الترجمة دقيقة , او متوسطة او بعيدة تماما مثل هذه النماذج .

→ Jane is visiting Africa in September.

→ Jane is going to be visiting Africa in September.

→ In September, Jane will visit Africa.

→ Her African friend welcomed Jane in September.

وبالتالي عملية قياس الاحتمالية probability هي لتكون اداة للخوارزم ليتمكن من الوصول لافضل النتائج , فكما في التوقع نقوم بفحص معادلة الخطأ بشكل مستمر للوصول لاقل قيمة لها , فهنا نقوم بفحص الاحتمالية للوصول لأعلي قيمة لها .

و قد يأتي سؤال , لم لا نستخدم تكنيك البحث الشره greedy search في الترجمة ؟
و هذا التكنيك يقصد به , ان يقوم الخوارزم باختيار اول كلمة بعناية , ثم يبحث ما هي الكلمة التالبة المناسبة للكلمة الاول فيختارها , ثم يبحث عن الكلمة التالية للكلمةالاولي و الثانية , وهكذا كلمة بكلمة .

و مشكلة هذا التكنيك انه لا يتناسب مع الترجمة , فلو كانت الجملة الفرنسية هي نفسهخا , وكان هناك نموذجين من الترجمة المتاح :

Jane is visiting Africa in September.

Jane is going to be visiting Africa in September.

النموذج الأول أكثر دقة لغويا , بينما النموذج الثاني هو ما قد يصل الخوارزم اليه اذا ما استخدم تكنيك البحث الشره , وذلك لان كلمة is تأتي غالبا ملحوقة بكلمة going لان تكرارها اكثر من كلمة visiting و بالتالي فالخوارزم سيتجه غالبا للنموذج الثاني وهو اقل كفاءة .

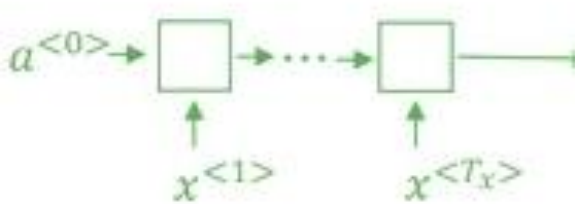
-

نتناول الآن تكتيك البحث الشعاعي beam search وهو الذي يستخدم في كثير من التطبيقات اللغوية .

فمثلا لدي ترجمة النصوص , او لدي تحويل الصوت الي نصوص , فنحن لا نريد من الخوارزم ان يقوم ببحث عشوائي و ايجاد اي كلمات (مثلما يحدث في تطبيق كتابة رواية او مقال عشوائي) , بل نريد كلمات محددة , والتي تعبر بالضبط عن ما نريده

و يمكننا تتبع مسار البيانات اثناء البحث الشعاعي كالتالي :

إذا كان لدينا الجملة الفرنسية المطلوب ترجمتها , فكما ذكرنا نقوم بإدخالها في الشبكة المركبة :



ولكن حينما يبدأ الخوارزم في انتاج المخرجات , يقوم أولا بتحديد أول كلمة في المخرج المناسب , اي اول كلمة مناسبة لبدأ الترجمة بها .

و يتم التحديد عبر استخدام سوفتماكس , من عشر الاف كلمة (عدد القاموس) لاختيار الكلمة ذات الإحتمالية الأعلي $P(y<1>)$.

و هنا علينا التعرف علي معامل مهم اسمه B وهو الذي تم تسميته بناء علي كلمة beam .

و يقصد به عدد الإختيارات المطلوب التعامل معها في نفس الوقت , فلو تم تحديد B = 3 , فهذا معناه اننا لن نختار فقط أعلي كلمة في السوفت ماكس بناء علي الإحتمالية , ولكن سنختار أعلي 3 كلمات .

فإذا كانت أعلي ثلاث كلمات في الإحتمالية , لتكون الكلمة الأول كترجمة للجملة الفرنسية هي :
(in , jane . september)

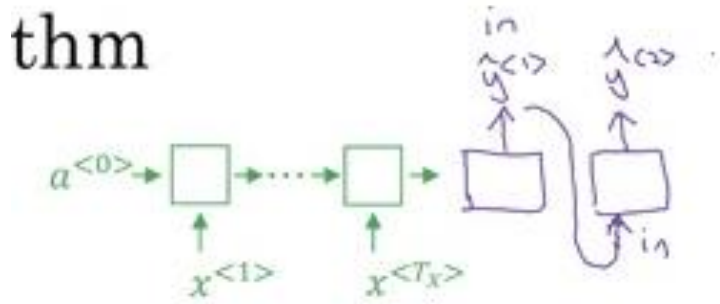
و كما قلنا , تم غختيارهم لأنهم أعلي قيم في الإحتمالية , بمعلومية الجملة الفرنسية , وهو ما يشار إليه P(y<1> | x)

ماذا عن الخطوة التالية ؟ هي البحث عن الكلمة الثانية في الجملة وهي y<2>

وقتها نقوم بتناول الكلمة الأولي (in) و تجريب أن يكون وراءها الكلمات العشر آلاف في القاموس , لتكون :

in a
in aaron
in book
..
..
in zulu

فيتم تجريب وضع الكلمات العشر آلاف ورائها , ثم حساب الإحتمالية الكلية لوجود الكلمتين معا . وهي ما يمكن أن نرسمها هكذا :



فبعد ان تم إخراج الكلمة الأولي in يتم استخدامها مع الجملة الفرنسية الاصل لمعرفة الكلمة الثانية , وهذا عبر حساب احتمالية الكلمة الثانية y<2> بمعلومية شيئين , الكلمة الأولي , و الجملة الفرنسية , وهي ما يمكن كتابته :

P(y<2> | x , y<1>)

هذه القيمة هي فقط احتمالية ايجاد الكلمة الثانية , لكن نحتاج الآن ، نقوم بحساب إحتمالية إيجاد الكلمة الأولي و الثانية معا , بمعلومية الجملة الفرنسية , والتي سيكون اسمها :

P(y<1> , y<2> | x)

و تكون قيمتها حسابيا , حاصل ضرب الإحتمالية الأولي (ايجاد الكلمة الأولي بمعلومية الجملة الفرنسية) , في الإحتمالية الثانية (إيجاد الكلمة الثانية بمعلومية الكلمة الأولي و الجملة الفرنسية) , و التي ستكون كالتالي :

P(y<1> , y<2> | x) = P(y<1> | x) * P(y<2> | x , y<1>)

فيكون لدينا 10 الاف احتمالية لكلمتين معا , لكن لن نختار منهم شيئا الآن , فلأن قيمة B تساوي 3 , و لدينا 3 كلمات تصلح ككلمة أولي , فسنكرر كل ما تم مع الكللتين الثانية و الثالثة :

jane a
jane aaron
jane book
..
..
jane zulu

و كذلك الأمر مع (September) .

فيكون لدينا 30 الف قيمة , لثلاث كلمات , كل كلمة تليها 10 الاف كلمة , وهنا لن نختار اكبر قيمة فيهم , بل أعلي 3 قيم لأنه هي قيمة الـ B , ولنتفرض أنهم :

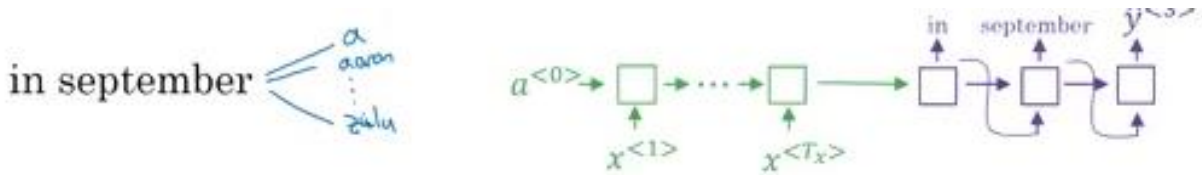
in september

jane is

jane visit

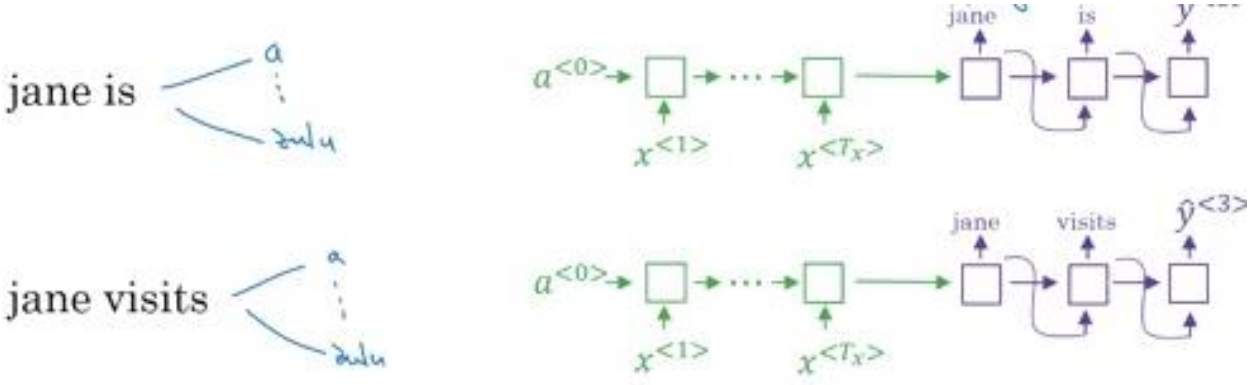
لاحظ أنه لا يشترط ان تكون الكلمات الثلاثة لهم بدايات مختلفة او متشابهة , قد تكون بداياتهم مختلفة او متشابهة , حسب قيم الاحتمالية .

ثم مرة اخري ينتقل للكلمة الثالثة , فيتناول الاختيار الأول : (in September) و يقوم بوضع خلفه الكلمات العشر الاف لقياس احتمالية كل كلمة فيهم هكذا



فنري في الجزء الأيمن decoder يقوم باستنتاج لكلمة الثالثة , بمعلومية الكلمتين الاولي و الثانية و الجملة الفرنسية . .

ثم يكرر الأمر في الجملتين الأخرتين :



فيكون لدينا ايضا 30 الف احتمالية , نقوم باختيار 3 منهم حسب الرقم B و نستمر في الكلمة الرابعة و هكذا .

و حينما يصل الخوارزم للكلمة <EOS> يتوقف هنا , ويقوم بقياس ايا من الاختيارات الثلاثة لها احتمالية اعلي و يقوم باخراجها .

-

سنتناول الان عدد من التعديلات في نظام البحث الشعاعي , لزيادة كفاءته

نبدأ بتسوية الطول length normalization .

و يقصد بها ان المعادلة العامة لإيجاد احتمالية تواجد جملة معينة من عدة كلمات , ستكون حاصل ضرب الإحتماليات معا كالتالي :

$$\arg\max_{y} \prod_{t=1}^{T_y} P(y^{<t>} | x, y^{<1>}, \dots, y^{<t-1>})$$

لكن المشكلة ان قيمة اي احتمالية هي بين الصفر و الواحد , و في حالة ضرب عدد كبير من القيم تقل عن الواحد , فالقيمة النهائية تتضائل بشدة حتي تقترب من الصفر , وبالتالي يصعب علي الخوارزم المقارنة بين القيم و بعضها .

فيكون البديل هو استنخدام اللوغاريتم للقيم , فبدلا من ضرب الاحتماليات في بعضها , نقوم أولا بإيجاد لوغاريتم ك حاصل الضرب معا , وهو ما يعني بالتبعية , مجموع لوغاريتمات الاحتماليات المنفصلة

$$\arg\max_{y} \sum_{y=1}^{T_y} \log P(y^{<t>} | x, y^{<1>}, \dots, y^{<t-1>})$$

و ذلك لأن :

$$\text{Log}\ (a * b * c * d) = \log a + \log b + \log c + \log d$$

ولأن دالة اللوغاريتم هي دائما دالة تصاعدية كالتالي :

فزيادة اللوغاريتم معناها زيادة القيمة الاصلية و هي الاحتمالية .

مع ملاحظة ان لوغاريتم اي رقم بين الصفر والواحد هو رقم سالب , فلوغاريتم الواحد يساوي 0 و لوغاريتم الصفر هو سالب مالا نهاية , فلوغاريتم 0.1 مثلا هو سالب 1 , ولوغاريتم 0.001 هو سالب 100

فتكون قيمة مجموع اللوغاريتمات معا هو قيمة سالبة كبيرة , ويتم الاختيار علي اكبر قيمة , اي اقربها للصفر , يتم اختيار مثلا -50 افضل من -70

كما أن هناك عيبا آخر سيظهر , وهو ان الخوارزم قد يميل ناحية اختيار جمل قصيرة الكلمات بدلا من الجمل الأكثر طولا , لان كلما زاد عدد الكلمات كلما قلت قيمة الاحتمالية , وهو ما سيجعل الخوارزم يميل للجمل القصيرة , حتي لو كانت الترجمة غير كافية او دقيقة .

و لحل المشكلة فيمكن ان نقوم بتعديل معادلة اللوغاريتم , حيث يتم قسمة القيمة علي Ty وهوعدد الكلمات الذي تم اختياره , وهي عملية التسوية normalization.

فلو كان تم اختيار 5 كلمات و كانت الإحتمالية الكلية -50 , او تم اختيار 8 كلمات و كانت الإحتمالية الكلية -64 , فلو تم القياس فقط بناء علي قيمة الإحتمالية فستكون الكلمات الخمس أفضل , بينما حينما يتم قسمة الـ -50 علي 5 , والـ -64 علي 8 , سنري ان الكلمات الثمانية افضل .

و أحيانا يتم وضع أس للـ Ty يسمي الفا , بحيث لو كان الاس يساوي صفر , فتختفي الـ Ty و نقوم بالغاء التسوية تماما , ولو كانت بـ 1 فتكون تسوية كاملة , و غالبا ما يتم اختيار رقم بين الصفر و الواحد , حتي تصير المعادلة :

$$\frac{1}{T_y^{\alpha}} \sum_{t=1}^{T_y} \log P(y^{<t>} | x, y^{<1>}, \dots, y^{<t-1>})$$

القيمة الأخري التي سنتتحكم في ضبط البحث الشعاعي هي قيمة B

لاحظ انه كلما زادت قيمة B كلما زادت الكفاءة و ذلك لأنه سيكون لديه خيارات أكبر , ولكن سيزيد الوقت , وكلما قلت الـ B كلما قل الوقت و الكفاءة معا . .

و اذا وصلت الـ B لرقم 1 . فهذا هو البحث الشره , الذي يتناسب فقط مع كتابة نصوص عشوائية , وليس مع الترجمة

فالرقم المعتاد للاستخدام للـ B هو 10 , احيانا يقل , واحيانا يزيد ليصل الي 100 , ونادرا ما يزيد ليصل للـ 1000 في التطبيقات المعقدة

-

كما درسنا من قبل في الكورس الثالث , عن تكنيك تحليل الأخطاء Error analysis, وفن معرفة أين تتواجد المشكلة الاساسية و فعلينا ان نستخدمها هنا في تكنيك البحث الشعاعي .

و ذلك لأن الخوارزم يحتوي علي خطوتين , خطوة الـ RNN و خطوة البحث الشعاعي , فعلينا معرفة من فيهما المتسبب في المشكلة لعلاجها

فلو كانت نفس الجملة الفرنسية , لها ترجمة دقيقة (ترجمة بشرية) هي :

Jane visits Africa in september

و كانت ترجمة الخوارزم :

Jane visited Africa last september

فسنسمي الأولي : y* و الثانية ŷ

فنريد أن نعرف سبب المشكلة , وهل هي من الـ RNN ام البحث الشعاعي . و ذلك لتحديد كيف نقوم بحل المشكلة , هل بعلاج الـ RNN ام بجمع المزيد من البيانات , ام بتغيير قيمة B ام الفا ام ماذا ؟

و قبل البدأ في تنفيذ تحليل الخطأ , علينا ان نتذكر شيئ هام , بالنسبة لتقسيم المهام بين الـ RNN و البحث الشعاعي .

الشبكة المتكررة RNN يكون دورها مختص بتوقع كلمات للترجمة المطلوبة , بينما البحث الشعاعي يركز علي تقييم كل كلمة و معرفة احتماليتها و اختيارها .

فكي نعرف سبب المشكلة , سنحسب قيمتي $P(\hat{y})$, $P(y^*)$, اي مقدار احتمالية الجملة الصحيحة , والجملة الغير صحيحة و الي ظهرت كنتيجة .

في حالة كان $P(\hat{y}) > P(y^*)$ أي أن احتمالية الجملة الصحيحة أكبر من الغير صحيحة , فهذا معناه أن الـ RNN قد قامت بدورها في استنتاج الكلمات المناسبة , لكن البحث الشعاعي قد عجز عن اختيارها بالتحديد و تقييمها بشكل مناسب , والوصول اليها , وقام هو باختيار جملة اخري غير مناسبة , وهذا معناه اننا نريد ضبط البحث الشعاعي .

بينما لو كانت $P(\hat{y}) < P(y^*)$ فهذا معناه أن الـ RNN من الاساس قد انتجت جمل غير مناسبة , في حين قام البحث الشعاعي بم عليه من اجل اختيارها و تقييمها , فعلينا علاج الشبكة المتكررة .

ويتم الفحص الكامل عبر عمل جدول مثل هذا , في عينة الاختبار او التطوير , واذا كان لدينا مثلا الف جملة في هذه العينة , فنقوم بوضع قيم $P(\hat{y})$, $P(y^*)$ في الجدول , وكتابة اما RNN او Beam في العمود الأخير

Human	Algorithm	$P(y^*\|x)$	$P(\hat{y}\|x)$	At fault?
Jane visits Africa in September.	Jane visited Africa last September.			

وبالتالي يمكن حساب ايهما له النسبة الأكبر من الأخطاء , ومعرفة المتسبب في المشكلة

-

حسنا , ماذا في حالة كان هناك أكثر من ترجمة دقيقة و جيدة للجملة ؟ ؟

فلو كانت لدينا الجملة الفرنسية :

Le chat est sur le tapis

فيمكن ان تكون ترجمتها :

the cat is on the mat

أو

there is a cat on the mat

فلتجنب اي تداخل , نستخدم تكنيك درجة بلو BLEU score , والتي هي اختصار لكلمات Bilingual Evaluation Understudy

و لبدأ التعامل مع الترجمة المزدوجة , علينا ان نقوم بتقييم الترجمة الناتجة من الخوارزم , لمعرفة مدي جودتها .

من العوامل التي نقوم بقياس مدي جودة النتيجة هي الدقة precision و هي ان نقوم بحساب مدي تواجد كل كلمة في كلمات الترجمة الناتجة , هل هي موجودة في الترجمة الحقيقية ام لا . .

لكن هذا المقياس قد لا يكون دقيق , فبفرض ان الخوارزم قد انتج ترجمة سيئة و هي :

the the the the the the the

فاذا قمنا بحساب الدقة , فسنجد ان كل كلمة ظهرت في الجملة الناتجة the متواجدة بالفعل في الترجمة الحقيقية سواء الاول او الثانية , وهو ما يجعل الدقة 7/7 و بالطبع هذا رقم غير سليم .

لذا نقوم بعمل تعديل في قانون الدقة , ان يكون البسط له حد اقصي , وهو عدد مرات تواجد الكلمة في الترجمة الحقيقية , و بالتالي ستكون الدقة 2/7 وهو رقم منطقي للترجمة السيئة .

و هذا التكنيك يسمي unigram اي الكلمة الواحدة .

بعد ان تعرفنا علي هذا مقياس الـ unigram فلنتعرف علي مقياس آخر وهو ما يسمي الكلمات الثنائية bigrams .

و يقصد بها ان تتواجد كلمتين متتاليتين معا , بين الترجمة الحقيقية , والناتجة .

فإذا كانت الترجمتين الحقيقيتين كما هما :

the cat is on the mat
there is a cat on the mat

و كانت الترجمة الناتجة هي :

the cat the cat n the mat

نلحظ انها لازالت سيئة لكنها افضل قليلا من السابقة .

فلحساب الكلمات الثنائية , نبدأ بتقطيع الجملة الناتجة الي كلمتين كلمتين , اي: الأول-الثانية , ثم الثانية-الثالثة , ثم الثالثة الرابعة , وهكذا ثم نقوم بحذف التكرار فيها , ثم قياس كم مرة ظهرت في الترجمة الناتجة , وكم مرة في احد التراجم الاصلية

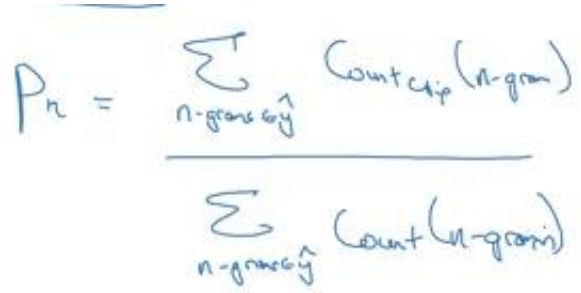
فيمكننا عمل الجدول التالي :

Bigrams	Count	Count-clip
the cat	2	1
cat the	1	0
cat on	1	1
on the	1	1
the mat	1	1

و بالتالي يكون BLEU bigram score يساوي 4/6 لأن العمود الاخير كان 4 , بينما المجموع 6 ثنائيات

وفي نفس الإطار يمكن حساب trigram للكلمات الثلاثية المتتالية , او quadgram للكلمات الاربع و هكذا .

و يكون القانون العام لدينا هو :



أي ان الاحتمالية هي مجموع ظهور الكلمات المتتالية بعد حذف التكرار في ايا من الترجمات الدقيقة , علي المجموع الكلي دون حذف .

و تكون الاحتمالية تساوي 0 حينما لا يكون هناك ايا من الكلمات متطابقة بشكل سليم و تكون 1 حينما تكون الترجمة الناتجة متطابقة تماما مع ايا من الترجمات الدقيقة .

و بالطبع كلما زاد عدد الكلمات المتتالية كلما كان التقييم ادق و اصعب , فقد تكون هناك ترجمة تحصل علي 0.7 في الـ bigram لكن هي نفسها علي 0.5 في الـ trigram و هكذا .

و يكون التقييم النهائي للـ BLEU score يساوي :

BLEU Score = Bp exp(¼ ∑ Pn)

أي أننا نقوم بحساب P1,P2,P3,P4 ,(حيث كل رقم يدل علي عدد الكلمات المتتالية) ونجمعهم و نقسمهم علي اربعة لايجاد المتوسط , ثم نقوم برفعهم للأس exp . واخيرا نضربهم في المعامل Bp

و هذا المعامل كي يتأكد أن الترجمة ستكون طويلة بما يكفي و لن تكون قصيرة للغاية , فالترجمة الطويلة يقل فيها الاحتمالية , فقيمة Bp تكون 1 اذا كانت الترجمة الناتجة اطول من الحقيقية , و تكون بقانون اخر معقد يجعلها اقل من 1 اذا كانت الترجمة الناتجة اقصر من الحقيقية

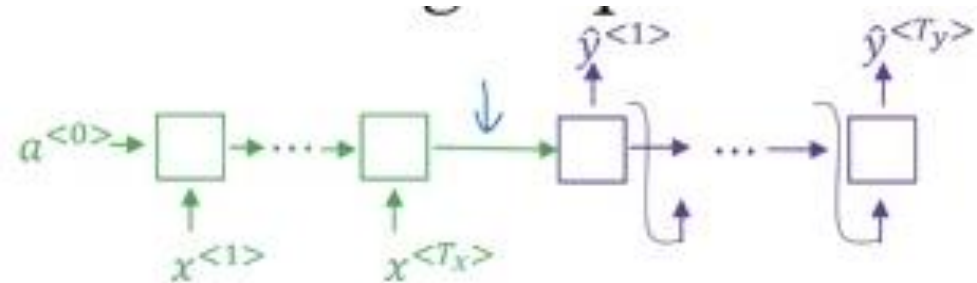
$$BP = \begin{cases} 1 & \text{if MT\_output\_length} > \text{reference\_output\_length} \\ \exp(1 - \text{MT\_output\_length}/\text{reference\_output\_length}) & \text{otherwise} \end{cases}$$

-

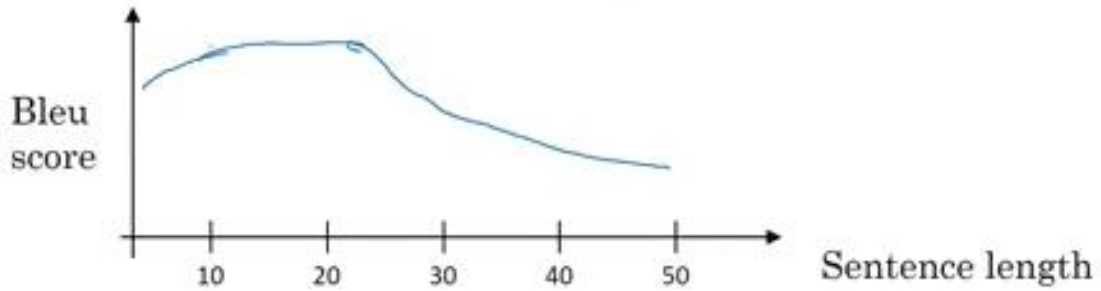
الأمثلة السابقة كلها كانت عن استخدام الـ RNN في ترجمة اللغات , حينما نقوم بتشفير encoding اللغة الأصلية , ثم تفسيرها decoding للغة الجديدة .

إلا أن هناك اسلوب يسمي نموذج الانتباه أو التركيز attention model و الذي يتم استخدامه مع الـ RNN لتفادي عدد من المشاكل, وأهمها مشكلة : ترجمة النصوص الطويلة .

فبفرض أن الجملة الفرنسية كانت 4 سطور , و هو ما يعني ان ترجمتها مرة واحدة ستكون صعبة , فلو قام المشفر encoder بتحليل عدد كبير من الكلمات , ثم قام المفسر decoder باستنتاج الكلمات , لن تنتج اي ترجمة دقيقة . .



حتي انه حينما نقوم بعمل رسم بياني بين طول الجملة المترجمة , وبين درجة بلو , نجد ان الدرجة تقل كثير مع الكلمات العديدة



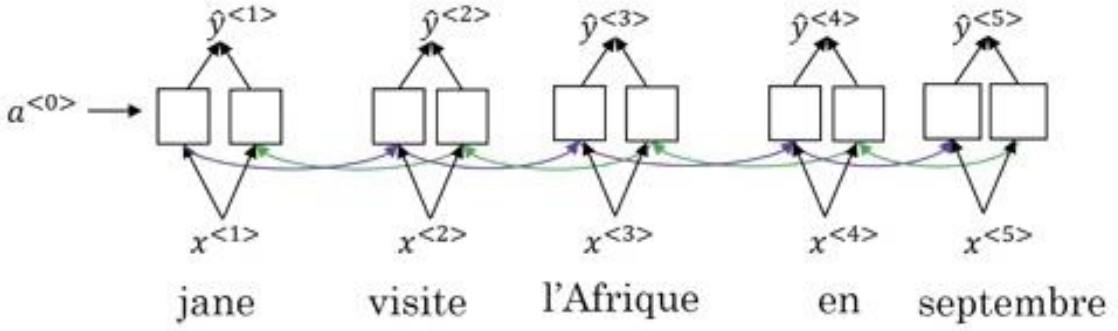
فالحل ان نتعامل معها كما يتعامل معها المترجم البشري , الإنسان الذي سيترجم جملة طويلة لن يحفظها في عقله ثم يترجمها , لكنه سيتناول جزءا جزءا منها ليقوم بترجمته بالترتيب .

ولكي نقوم بهذا الأمر سنجعل الخوارزم نفسها حينما يقوم باستنتاج كل كلمة من كلمات الترجمة الخارجة (الإنجليزية), سنجعله يركز علي الكلمات الداخلة (الفرنسية) بكميات و نسب مختلفة .

فمثلا لو كان الخوارزم الان يستنتج الكلمة الخامسة من الترجمة الإنجليزية , فيمكن أن يركز علي الكلمة الرابعة نسبة كبيرة (30%) و الثالثة و الخامسة بنسبة اقل (15 % لكل منهم) , والاولي و الثانية و السادسة أقل (5 %) , وهكذا , بحيث يكون مجموع التركيز 100 % .

و هذا الأمر يجعل استنتاج اي كلمة انجليزية يكون بناء علي التركيز علي عدد محدد من الكلمات الفرنسية , وتجاهل كلمات اخري قل اهمية في التاثير عليها .

فإذا كان لدينا نفس الجملة الفرنسية , ونريد أن نترجمها , فيمكن في البداية ان نجعل كل كلمة من الكلمات الفرنسية (الإكسات) تدخل في شبكة متكررة ثنائية هكذا Bidirectional RNN



و بالتالي حينما قوم الـ RNN بمرحلة التفسير decoding لاستنتاج الكلمات الإنجليزية (بعد انتهاء مرحلة التشفير encoding الفرنسية) فتكون أول خطوة اي استنتاج كلمة Jane لتكون أول كلمة إنجليزية y<1> .

و هنا علينا ان نتعرف علي المعامل ألفا α

وهو المعامل الذي سيخبر الخوارزم , لي من يركز بالتحديد خينما يقوم باستنتاج الكلمات الإنجليزية .

فللكلمة الإنجليزية الأولي s<1> سيكون لها عدد من معاملات الفا مثل :

α (1,1) , α (1,2) , α (1,3) , α (1,4) ,

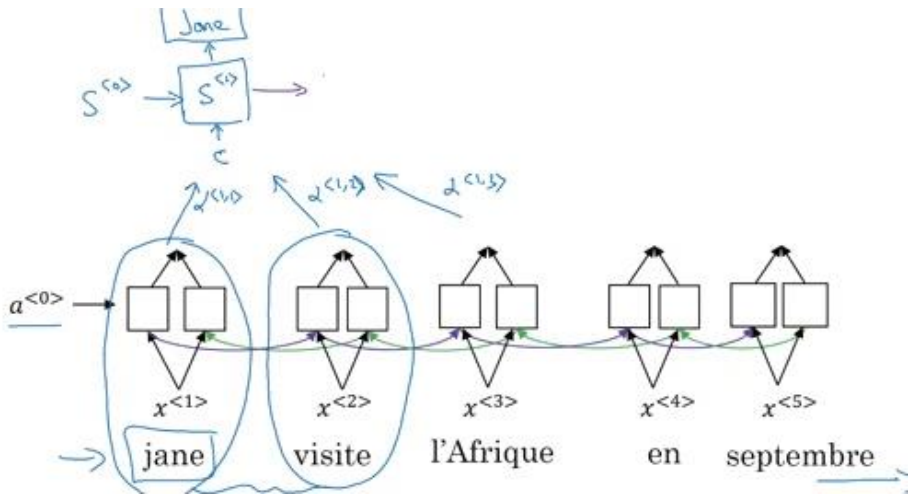
حيث الرقم الأول يشير إلي الوحدة المستقبلة للغة الإنجليزية , أما الرقم الثاني , فهو عن مدي تركيزها في الكلمات الفرنسية

و جميع قيم الفا لوحدة ما , تتروح بين الصفر و الواحد , و مجموع كل قيم الفا لوحدة ما يساوي 1 صحيح .

و بالتالي كمدخل للوحدة الأولي التي ستنتنتجها , سيكون هناك مدخل لها يسمي c وهو اختصارا للـ context و الذي يساوي :

$c^{<1>} = \alpha^{(1,1)} * a^{<1>} + \alpha^{(1,2)} * a^{<2>} + \alpha^{(1,3)} * a^{<3>}$ . .

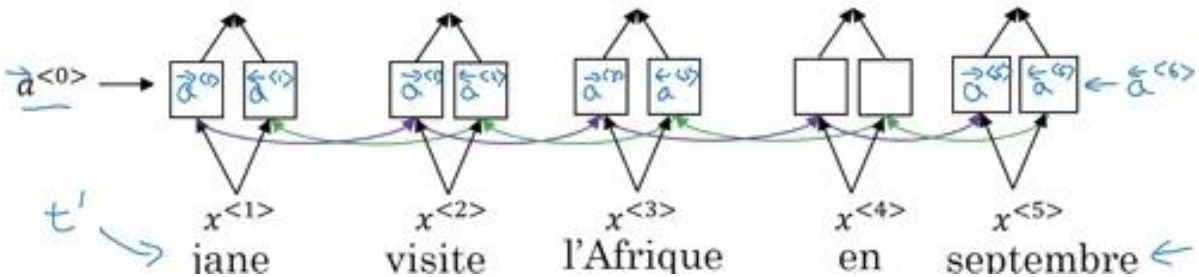
والتي تكون بالرسم :



حيث قيم a هي دالة الـ activation من الكلمات الفرنسية , و هذه الـ c<1> هي التي تدخل في استنتاج الكلمة الأولي . .

وهو ما سيجعل لدي استنتاج كل كلمة انجليزية , يتم التركيز علي كلمات محددة , وبنسب مختلفة حسب أهميتها لها

ولا نريد أن ننسي أن شبكة التشفير encoding للجملة الفرنسية هي شبكة ثنائية الاتجاه bidirectional RNN لذا فإنها فعليا لها قيمتي `a واحدة يمين (مسار أمامي) و الثانية يسار (مسار خلفي)



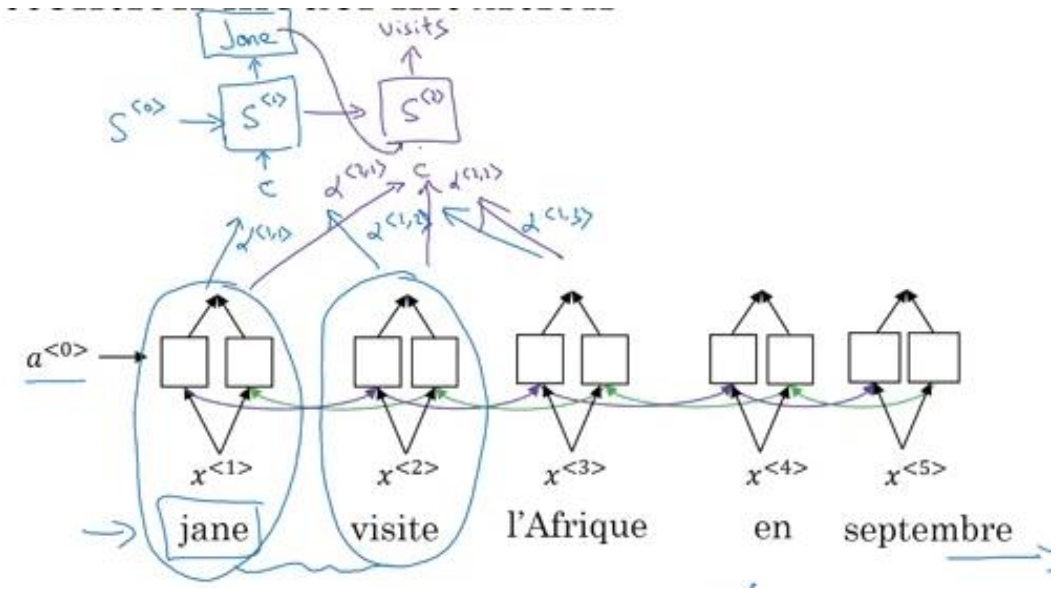
و تكون قيمة `a النهائية هي محصلة بينهما

$$a^{<t>} = (\vec{a}^{<t>}, \overleftarrow{a}^{<t>})$$

لا تنس أنها كلها برايم ` لأنها مدخلات

مثلها حينما نقوم باستنتاج الكلمة الثانية سيكون هناك اوزان , لتجعلها تركز علي الكلمات الفرنسية بنسب مختلفة , بالإضافة لاعتمادها علي الكلمة الإنجليزية الأولي :

α (2,1) , α (2,2) , α (2,3) , α (2,4) ,



و أيضا :

c<2> = α (2,1) * a<1> + α (2,2) * a<2> + α (2,3) * a<3> . .

فتكون القيمة العامة لألفا بالشكل :

$$\alpha (t , t`)$$

حيث t` فوقها شرطة , وتنطق تي برايم هي خاصة بالكلمة الفرنسية (المدخلات), اما الـ t فهي الإنجليزية (المخرجات)

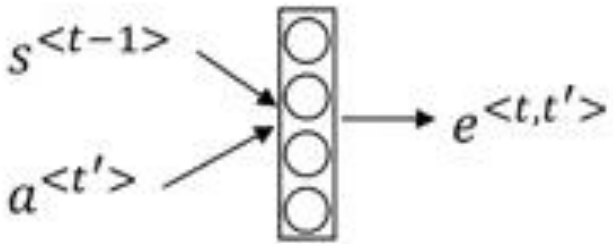
ويكون معناها : مقدار الإهتمام او التركيز المفترض إعطائه اثناء قراءة الكلمة المدخلة **t`** لاستنتاج الكلمة المخرجة **t**

$$\alpha^{<t,t'>} = \text{amount of attention } y^{<t>} \text{ should pay to } a^{<t'>}$$

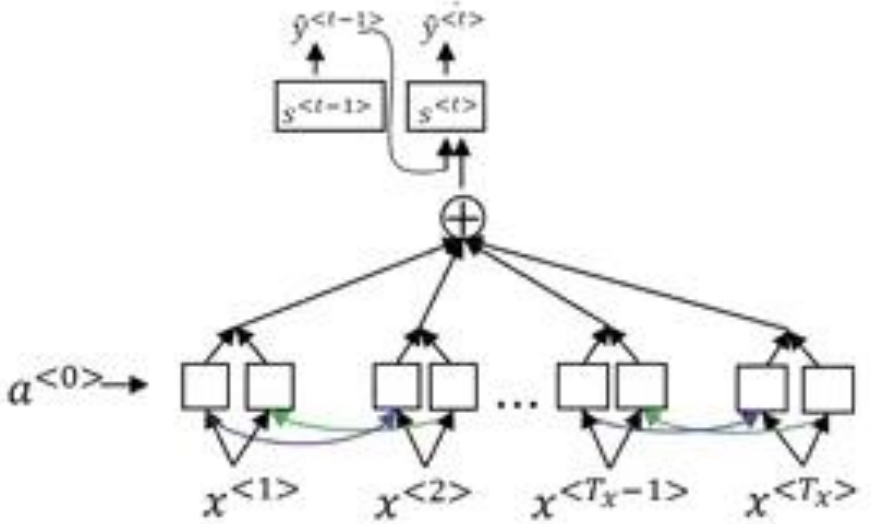
و لحساب قيمة ألفا معينة , تكون بالقانون :

$$\alpha^{<t,t'>} = \frac{\exp(e^{<t,t'>})}{\sum_{t'=1}^{Tx} \exp(e^{<t,t'>})}$$

حيث قيمة e المذكورة , تتم عبر تطبيق شبكة NN بسيطة , يكون المدخل فيها هي قيمة s<t-1> و a<t`> لأنهما بشكل ما هما الذان يتحكمان في قيمتها :



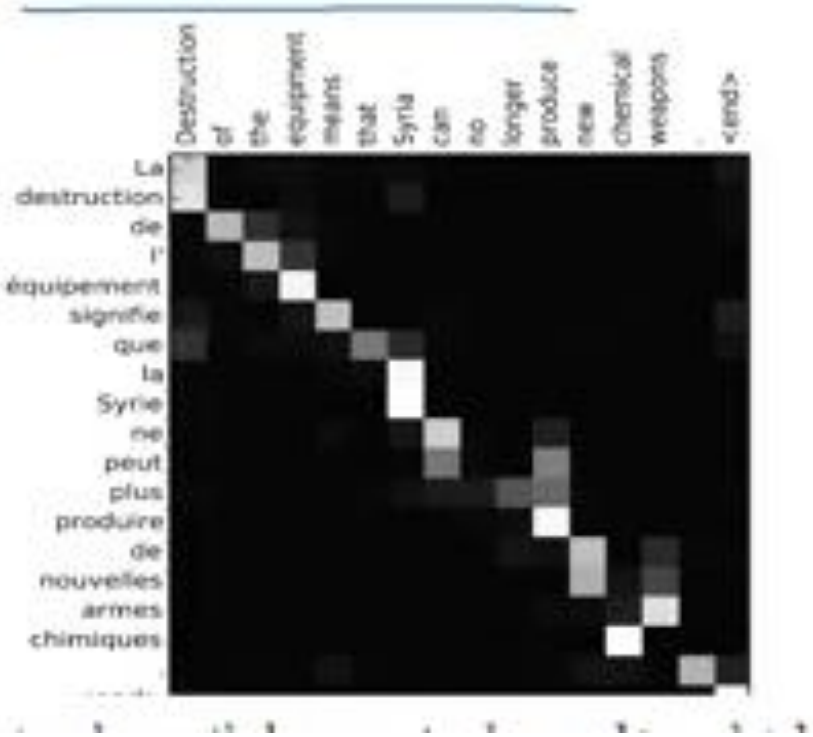
فيكون الشكل العام :



و عيبها الخطير الوقت الطويل بها , فلو كان لديك 20 كلمة مدخل و مثلها مخرج , فسيكون هناك 20 في 20 عملية مضاعفة في الوقت .

و من تطبيقاتها كذلك , توصيف الصور image captioning لأنها تجعل الخوارزم يركز علي اجزاء معينة من الصورة ., ليتعرف عليها و يكتب وصفا لها

و هنا رسم بياني يقوم بتوضيح الكلمات الفرنسية في المحور الرأسي و الانجليزية في الأفقي . و ما يوازيها من أوزان


Destruction
of
the
equipement
means
that
Syria
can
no
longer
produce
new
chemical
weapons
<end>
La
destruction
de
l'
équipement
signifie
que
la
Syrie
ne
peut
plus
produire
de
nouvelles
armes
chimiques

-

التطبيق الأخير في المنهج هنا : هو تحويل الصوت إلي نصوص audio recognition

فالصوت هو عبارة عن خلخلة في طبقات الهواء و اختلافات معينة في الضغط , مما يجعل الأذن لديها القدرة علي ماع الصوت و ترجمته الي معاني محددة . .

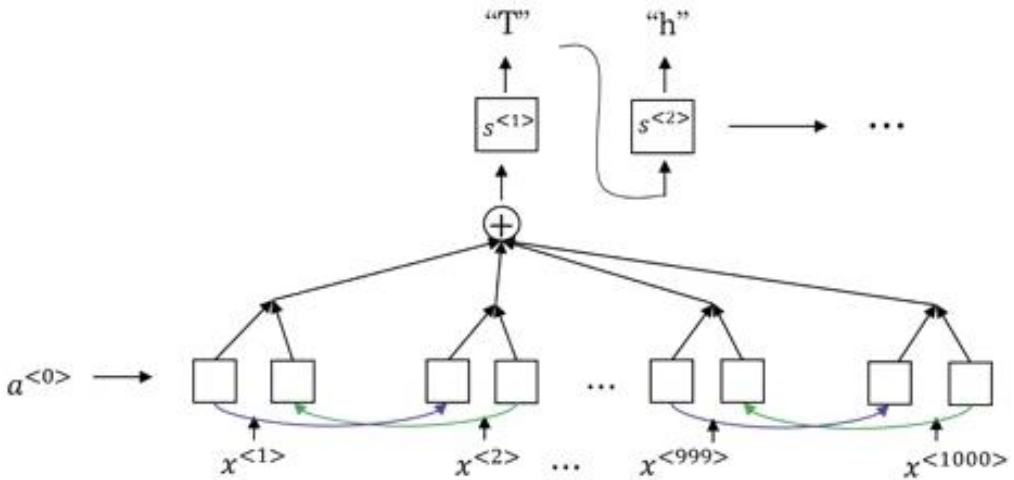
فالصوت يكون هكذا :

حيث المحور الأفقي هو خاص بالزمن , بينما المحور الرأسي يعبر عن مقدار حدة الصوت .

و قديما , حاول علماء اللغويات والصوتيات , فك اي كلمة إلي مقاطع صوتية محددة , وتسمي phonemes فكلمة python يتم فكها لثلاث اجزاء : pi + tho + on

إلا أنه مع تطور الشبكات العميقة , فاستخدام المقاطع الصوتية لم يعد مجديا , خاصة مع الكميات الهائلة من بيانات التدريب التي تصل أحيانا لـ 100 ألف ساعة صوتية , ولها ما يقابلها من النصوص المكتوبة .

وتكون الشبكة مشابهة لتلك المستخدمة في الترجمة السالف شرحها , بحيث تتناول الصوت , وتقوم بإنتاج الحروف او الكلمات تباعا



و هنا خاصية مهمة لابد من الانتباه لها .

مدخلات الصوت يكون التغير في التردد الصوتي , و إذا كان الصوت مسجل علي تردد 100 هيرتز , اي 100 قيمة في الثانية .

فلو كان لدينا 10 ثواني , فهذا معناه 1000 قيمة في المقطع الصوتي , فعدد المدخلات هو 1000 في الشبكة العصبية .

بينما عدد المخرجات (عدد الحروف) سيكون أقل بكثير , ففي الثواني العشر قد يكون مجموع الحروف الكلي هو 20 حرف مثلا.

هنا لابد من استخدام تكنيك CTC cost و الذي هو اختصار connectionist temporal classification .

و يقصد بها ان جملة بسيطة مثل :

the quick brown fox

سيكتبها الخوارزم في البداية

ttt_h_eee_____␣____qqq__uuu . . .

علي اعتبار ان هناك تكرارات لابد منها , وأن كل رموز underscore (_) تعبر عنالفواصل بين الحروف , بينما الرمز ␣ يعبر عن المسافة الحقيقية بين الكلمات .

و بعدها يتم تحويل هذه الحروف الي كلمات حقيقية

-

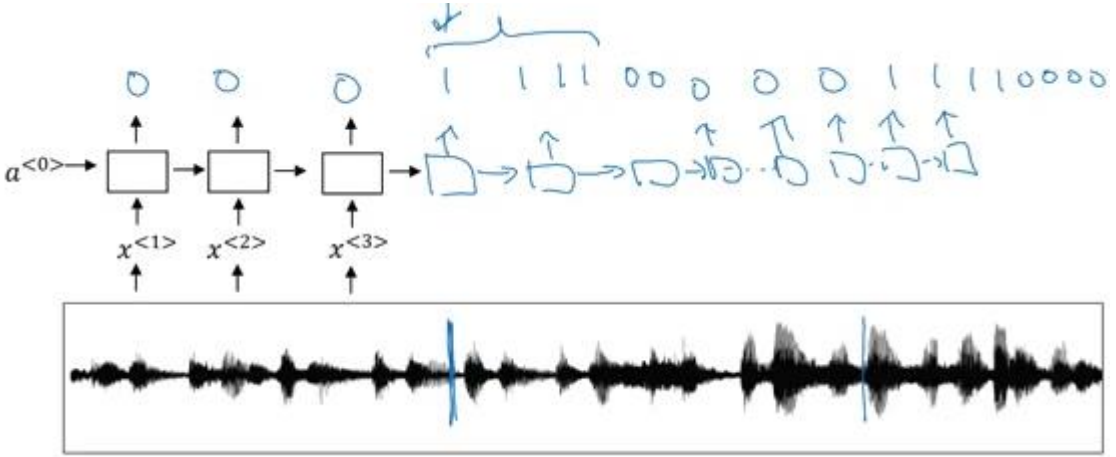
نصل إلي التعامل مع الكلمات الافتتاحية trigger word , وهي الكلمات التي تستخدم لايقاظ الجهاز التفاعلي من حالة السبات ليبدأ في الانتباه الي ما ستقوله .

Amazon Echo (Alexa) | Baidu DuerOS (xiaodunihao) | Apple Siri (Hey Siri) | Google Home (Okay Google)

و هناك عدد من التكنيكات المستخدمة من الكلمات الإفتتاحية, نتناول منها هذه الفكرة :

أن نقوم ببناء شبكة RNN , وعبر استخدام supervised algorithm , أن نقوم بدخال مقطع صوتي كامل فيه كلمات عديدة , ثم فيه الكلمة الإفتتاحية ok google في مكان معين , وأن نقوم بجعل المخرج بالكامل اصفار , عدا الجزءالمذكور فيه الكلمة الإفتتاحية , ان يكون بقيمة 1 , فهذا يجعل الخوارزم يتدرب عليها جيدا .

و يفضل أن يتم تكرار 1 اكثر من مرة , و ذلك لان الكلمة الافتتاحية ستستغرق مدة لن تقل عن ثانية , فيتم عمل ارقام 1 لما يتوازي معها



-

نهاية الاسبوع الثالث و الكورس بالكامل