ترکیب پردازش زبان طبیعی و بینایی ماشین در هوش مصنوعی

(پروژه توصیف تصاویر)

نویسنده: محمد تقی زاده

1402-1403

فهرست مطالب

چکیده

پردازش زبان طبیعی یکی از مباحث بسیار پرچالش و پرکاربرد در دنیای امروز است که از مفاهیم بین رشته ای بین دانشمندان علوم داده و کامپیوتر و زبان شناسان محسوب میشود ، هدف این پروژه توصیف تصویر با استفاده از مفاهیم جدید حوزه NLP است که در مقالات به آن ها پرداخته شده است. همانظور که میدانیم توصیف تصویر در دو حوزه بینایی ماشین و حوزه پردازش زبان طبیعی قرار میگیرید و به همین منظور برای این فیلد باید هم با مفاهیم بینایی ماشین آشنا باشیم نظیر شبکه های کانولوشنی و هم مفاهیم پردازش زبان طبیعی برای تولید متن خروجی که در پیاده سازی آن نیاز به آشنایی با مدل های زبانی داریم. رویکرد ما در این پژوهش براساس جدیدترین مبحثی که در NLP مطرح است یعنی مکانیزم توجه و مکانیزم Transformer ها خواهد بود که در سال ۲۰۱۷ توسط گوگل معرفی شدند و انقلابی در دنیای NLP بود و پس از آن مقاله بسیار مهم و تاثیر گذار Vision Transformer (ViT) که هدف آن استفاده از مکانیزم توجه ترنسفورمر ها در دنیای بینایی ماشین بود. رویکرد های پردازش زبان طبیعی پیش از ظهور ترنسفورمر ها با شبکه های عصبی بازگشتی یا RNN ها شروع شد که با اضافه کردن یک عنصر داخلی به شبکه های feed forward بعنوان hidden state سعی داشتند که state درونی عناصر یک داده دنباله ای را ذخیره کنند ، در ادامه چالش های اساسی این شبکه ها و تئوری های ریاضی پشت آن ها را دیدیم ، نظیر چالش بسیار حاد محو گرادیان که شبکه های بازگشتی روی داده های دنباله ای و سری های زمانی بسیار مستعد آن هستند. مهمترین بهبود شبکه های RNN ، شبکه های LSTM هستند به معنی شبکه های دارای حافظه های کوتاه مدت طولانی که سعی میکنند دنباله هایی معنا دار تر بسازند و جریان ورودی اطلاعات به حافظه یا cell state را با گیت های درونی خود کنترل کنند تا چالش های محو گرادیان و وابستگی های طولانی مدت در داده های دنباله ای را حل کنند . پس از این مراحل شبکه های بازگشتی و شبکه های LSTM را در فیلد هایی نظیر مدلسازی یک زبان و پیکره متنی ، تحلیل احساسات از متن و ترجمه ماشینی بررسی کردیم. ذات شبکه های بازگشتی مشکلاتی نظیر عدم قابلیت موازی سازی محاسبات را ایجاد میکند و همچنان با وجود شبکه های LSTM هم مشکل محو گرادیان و وابستگی های طولانی مدت در دنباله ها وجود دارد و در انتهای بخش ، مکانیزم توجه را بررسی کردیم که این مکانیزم سرآغازی بر مکانیزم ترنسفورمرها شد. در این بخش معماری Transformer ها و مقاله بسیار مهم attention is all you need که یکی اثرگذارترین مقالات فیلد NLP است را بررسی کردیم و دیدیم که چگونه این معماری با ارائه مفهوم self-attention درون decoder و encoder ها شیوه کار را به کل تغییر داد و مشکلات شبکه های بازگشتی را حل کرد.

پس از بررسی مکانیزم ترنسفورمر ها به هدف و انگیزه اساس پژوهش یعنی استفاده از مکانیزم ترنسفورمر در بینایی ماشین و توصیف تصویر رسیدیم. رویکرد مقاله ViT (که توسط گوگل معرفی شد) سعی میکند تا عملکرد ترنسفورمر ها را برای استخراج ویژگی های معنا دار از تصویر دیدیم بطوریکه مکانیزم توجه در بحث تصویر دقیقا مشابه متن به ما کمک میکند تا اجزای معنا دار یک تصویر را پیدا کنیم و با این اجزای معنا دار میتوانیم تسک های image classification ، image segmentation ، image captioning ، object detection و غیره را انجام دهیم. در انتهای کار نیز پروژه توصیف تصاویر را با استفاده از Vision Transformer ها در کتابخانه transformers پایتون و بستر hugging face پیاده سازی کردیم که کد های آن بصورت عمومی در گیتهاب در دسترسی است[12].

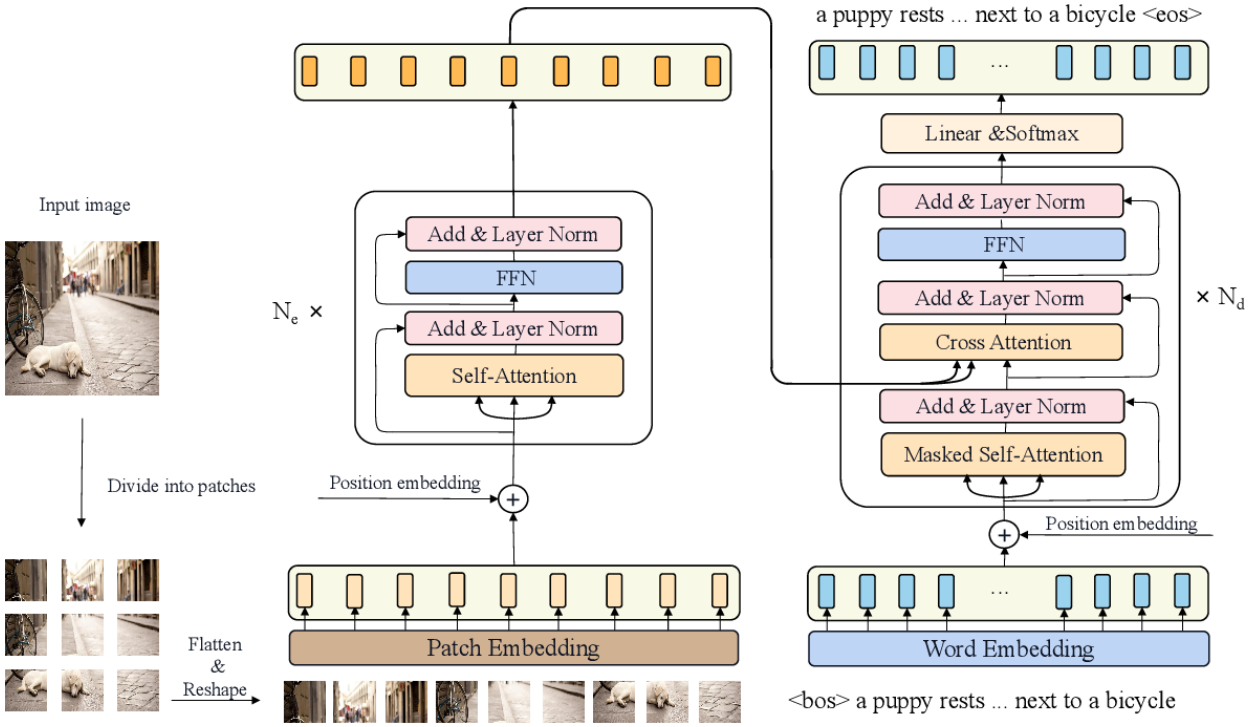


Figure ۱ : CPTR: Full Transformer Network for Image Captioning https://arxiv.org/pdf/2101.10804.pdf

بخش اول: مقدمه ای بر عملیات بینایی ماشین و شبکه های عصبی عمیق

عملیات مرسوم در بینایی ماشین

۱- دسته بندی تصاویر (Image classification) : مهمترین و مشهورترین تسک بینایی ماشین است ، در این تسک یک تصویر وارد میشود و مدل ما به این تسک یک کلاس اختصاص میدهد و محوریت کار ما و رویکرد مقاله Vision Transformer گوگل مبتنی بر همین تسک است.

۲- قطعه بندی تصاویر (Image segmentation) : در این تسک برای قطعه بندی معنایی ، هر پیکسل از تصویر کلاس بندی میشود و عملا مرز کامل شی مورد نظر استخراج میشود و این روش در مباحثی که object detection بصورت معمول دقت لازم را حاصل نمیکند استفاده میشود ، نظیر مباحث پزشکی که نیاز است کاملا توده و پیکسل های آن تصویر مرزبندی شود.

۳- تشخیص اشیا (object detection): در این تسک هم عملیات localization و پیش بینی bounding box دور شی انجام میشود و هم کلاس شی تشخیص داده میشود.

۴- توصیف تصویر (Image captioning): محوریت کار ما در این پژوهش روی این تسک است و هدف از آن این است که یک تصویر وارد میشود ، مدل باید به بخش های معنادار این تصویر توجه کند و در نهایت یک متن مناسب توصیف این تصویر تولید کند.

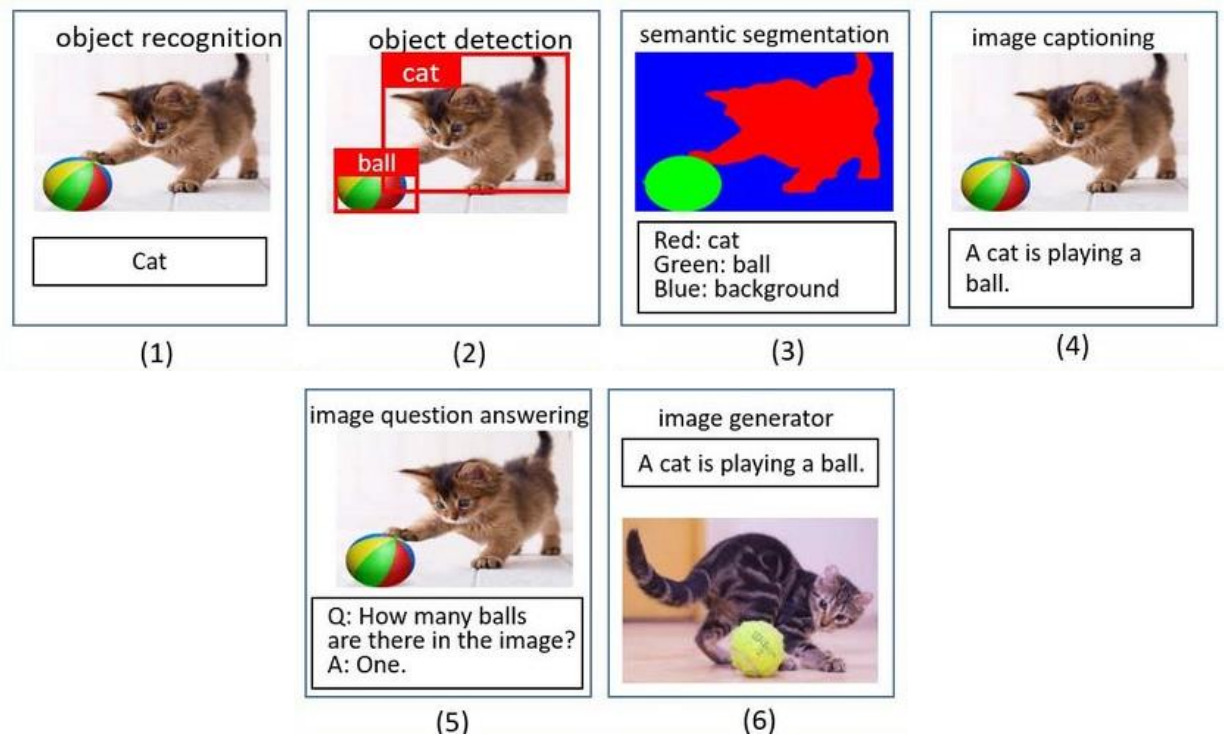
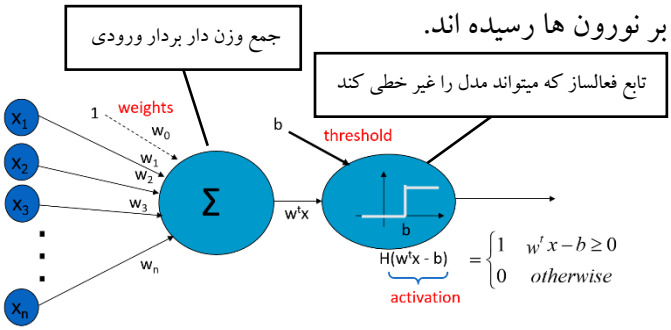


Figure ۲ : https://www.researchgate.net/figure/The-differences-among-six-popular-computer-vision-tasks-1-Object-recognition-sometimes_fig1_335013237

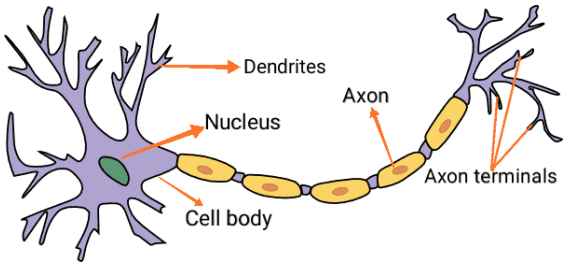
انواع شبکه های عصبی رایج و کاربرد های آنها

برای درک شبکه های عصبی بازگشتی باید با شبکه های عصبی معمولی و انواع مختلف آن ها را نیز بشناسیم و با آن ها آشنا باشیم که در این پروژه این مورد پیش زمینه پژوهش در نظر گرفته شده است ولی برای ساختارمند تر شدن بحث ، در ابتدا ساختار شبکه های عصبی معمولی را نیز مبصورت اجمالی معرفی میکنیم.

نورون های بیولوژیکی و مصنوعی : شبکه های عصبی با تقلید عملکرد مغز انسان سعی میکنند مشابه انسان درک کنند و با الهام از ساختار مغز انسان به یک مدل مبتنی بر نورون ها رسیده اند.

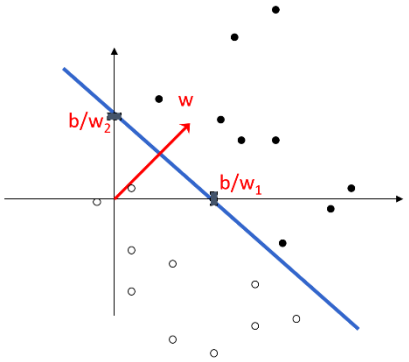


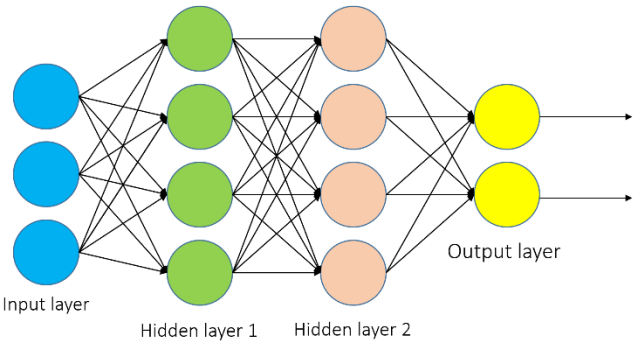
b) نور



a) نورون های بیولوژیکی انسان

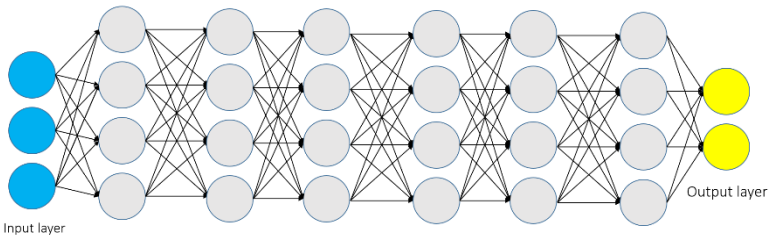
۱- **Feed forward neural network**: شبکه های عصبی مرسوم در یادگیری عمیق که پیش از این با آنها آشنا بوده ایم و از یک لایه ورودی یک یا دو لایه مخفی و یک لایه خروجی تشکیل شده اند و به این شکل کار میکنند که ابتدا ورودی را دریافت میکنند ، بعد از عبور مقادیر بردار ورودی از لایه های مخفی و اعمال وزن ها روی آن ها به خروجی میرسند ، در این شبکه ها هیچگونه دور یا سیکلی در وسط شبکه روی لایه ها نداریم و در انتهای کار پس از انتشار وزن های رندوم و غیر دقیق در شبکه (در ابتدا) و در انتها محاسبه خطای پیش بینی مدل با loss function های مختلف در نهایت براساس الگوریتم گرادیان کاهشی (gradient descent) ، خطا به لایه های قبلی بازانتشار یا Back propagate میشود و وزن های هر لایه به نسبتی و با یک Learning rate مشخصی که در طول زمان نیز تغییر میکند ، آپدیت میشوند که عملا بخش اساسی فرآیند یادگیری ، همین آپدیت پارامتر اصلی شبکه یعنی وزن ها توسط الگوریتم گرادیان کاهشی است که با عملکرد این الگوریتم آشنا



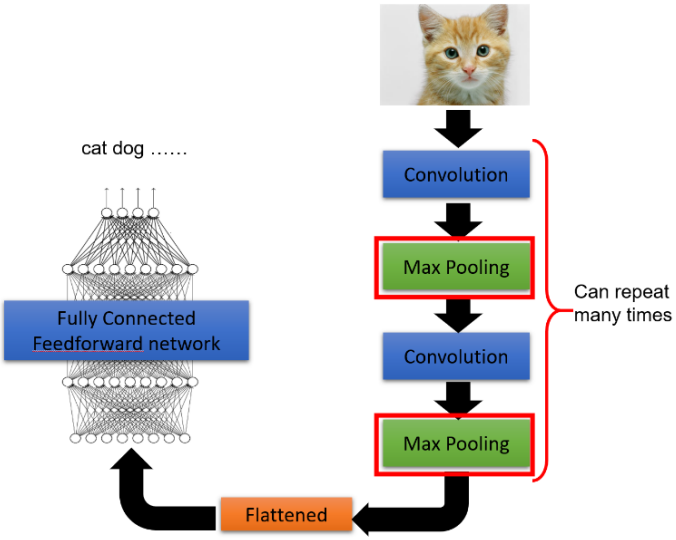


هستیم و در بخش بررسی عمیق شبکه های بازگشتی عملکرد آن و چالش هایی که ممکن است بوجود آید نظیر چالش های محو و یا انفجار گرادیان را بررسی میکنیم. این شبکه ها ساختاری بسیار ساده دارند و میتوانیم از آن ها برای مدل کردن یک فضای مساله ساده بصورت خطی نظیر مدل مقابل برای classification دو کلاس با فضای مساله ساده استفاده کنیم.

۲- **Deep neural network**: ساختار شبکه های عمیق همان شبکه های اولیه مصنوعی است با این تفاوت که در این شبکه ها چندین لایه مخفی (حداقل بیش از ۲ لایه مخفی) داریم که این لایه های مخفی بیشتر و عمق بیشتر باعث میشود مدل هایی بسیار قوی تر با قابلیت یادگیری خیلی قوی تر برای مسائل پیچیده تر داشته باشیم.

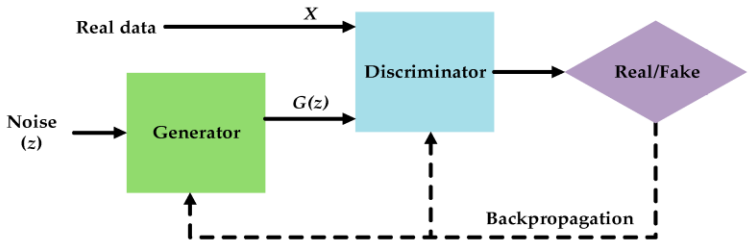


۳- **Convolutional neural network**: شبکه های عصبی کانولوشنی به شدت مناسب برای بینایی ماشین هستند و ساختار آن ها بصورت خلاصه به شرح شکل مقابل است : بین چیزی که انسان با دیدن تصویر درک میکند با چیزی که کامپیوتر و مدل بینایی ماشین برای استخراج ویژگی های مناسبی از تصویر که مفهوم تصویر را پرزنت کند پیش از یادگیری عمیق در الگوریتم های کلاسیک پردازش تصویر یک شکاف معنایی یا Semantic gape وجود داشت و برای استخراج کرنل های مناسب تکنیک ها و الگوریتم های مختلفی توسعه داده شده بود که همگی باز هم این شکاف معنایی را داشتند ولی در یادگیری عمیق با اعمال لایه های کانولوشن شبکه CNN بصورت Unsupervised Feature(Kernel) Learning ، کرنل ها را آموزش میبینند و این بسیار فرآیند یادگیری مدل بینایی ماشین را بهبود میدهد.



۴- **Generative adversarial networks** [11]: شبکه های GAN در چند سال اخیر از پرکاربردترین شبکه های عصبی هستند. در یادگیری عمیق چالش همیشگی محققین این بوده است که شبکه های مبتنی بر یادگیری عمیق برای آموزش بهتر به تعداد بیشتری مجموعه داده نیاز دارند تا بتوانیم لایه های شبکه را افزایش دهیم و با بالارفتن تعداد پارامتر های شبکه عمیق ، شبکه overfit نشود. شبکه های GAN با تولید نمونه های فیک واقعی به شدت مناسب این عملیات هستند که در دیتاست های کوچک بتوانیم نمونه های فیک بسازیم که بسیار واقعی به نظر میآند. کارکرد این شبکه ها بصورت خلاصه مشابه شکل زیر است ، در این شبکه ها دو بخش اساسی داریم

Generator: نمونه های فیک را تولید میکند.

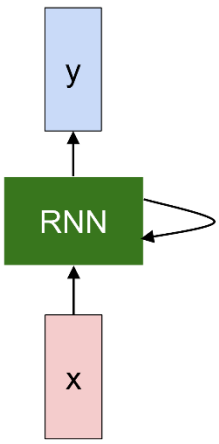


Discriminator: واقعی بودن یا فیک بودن تصویر تولید شده را براساس دیتاست واقعی تشخیص میدهد و Generator به این شکل سعی میکند تصاویر یا نمونه های واقعی تولید کند.

از کاربردهای این شبکه تولید تصاویر واقعی و نمونه های واقعی از روی دیتاست است و کاربرد دیگر آن تولید تصویر از روی یک متن است (دقیقا برعکس Image captioning).

۵- **Recurrent neural network:** شبکه های عصبی بازگشتی برخلاف شبکه های feed forward که همان شبکه های معمولی هستند دارای یک بخش hidden state هستند که عملا حافظه این شبکه ها محسوب میشود و بحث اساسی ما در پردازش زبان طبیعی با بررسی این شبکه ها آغاز میشود و چالش هایی که دارد را بررسی میکنیم و مدل های جدیدتر و مقالات جدیدتر این فیلد نظیر Transformer ها و مکانیزم Attention را بررسی میکنیم.

کاربرد شبکه های عصبی بازگشتی : در هر مجموعه داده ای که یک سری داده مرتبط به هم داریم (به عبارتی داده های Time series) شبکه های عصبی بازگشتی کاربرد دارند.



- توصیف تصاویر یا Image captioning
- پیش بینی سری های زمانی و داده های Time series : نظیر تحلیل بازار های مالی
- پردازش زبان طبیعی یا NLP
  - مدل سازی زبانی یا Language modeling
  - تحلیل احساسات و دسته بندی متن
  - ترجمه ماشینی و Machine translation
  - تحلیل صوت و گفتار
  - و....

بخش دوم : کارهای مرتبط پیشین در فیلد توصیف تصویر و پردازش زبان طبیعی

در فیلد پردازش زبان طبیعی پیش از معرفی مکانیزم ترنسفورمر ، شبکه های RNN و LSTM ها بسیار مورد استفاده و مطرح بودند و در زمینه توصیف تصاویر نیز از شبکه های LSTM با عنوان معماری های End to End یا Encoder to Decoder یا Sequence to Sequence استفاده میشد بنابراین پیش از بررسی مقاله های Attention is all you need که ترنسفورمر ها در آن توسط گوگل معرفی شد و سپس مقاله Vision Transformer که استفاده از ترنسفورمرها در بینایی ماشین توسط گوگل مطرح شد ، باید با این شبکه های پیشین که زیربنای این حوزه را ساخته اند نیز آشنا شویم.

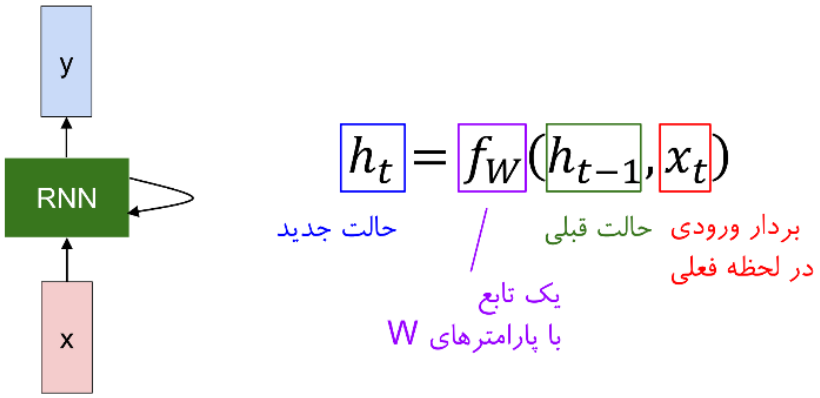
۲-۱ شبکه های بازگشتی یا RNN [5]

چرا شبکه های Feed forward مناسب داده های سری زمانی یا دنباله ای مثل جملات و کپشن ها نیستند؟ [10]

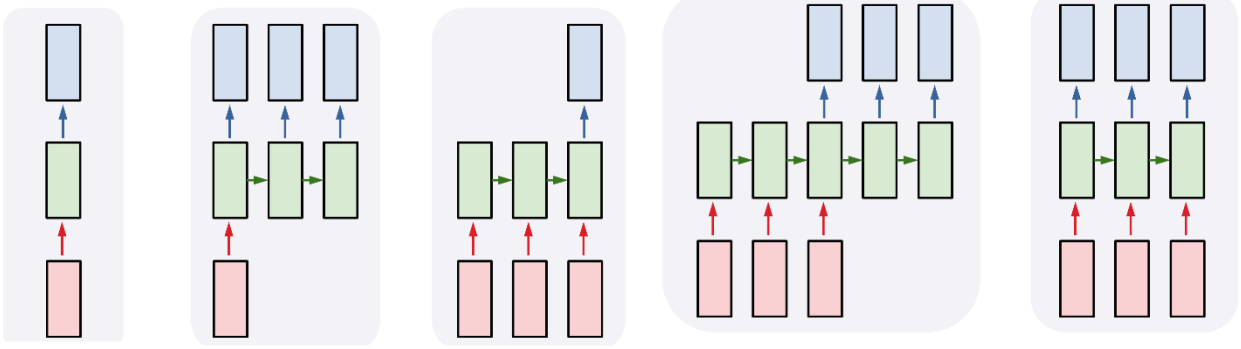
شبکه های عصبی بازگشتی ، همان شبکه های های Feed forward هستند با یک تفاوت مهم یعنی حافظه ، که دقیقا تفاوت اصلی و بهبود اصلی در همین بخش حافظه صورت گرفته است و باعث شده است که شبکه های عصبی بازگشتی مناسب داده Time series شوند (نظیر متن که دارای یک سری کلمات پشت سرهم است که مفهومی را در قالب یک جمله ارائه میدهد). زیرا که شبکه های Feed forward معمولی از آنجایی که سیکل در داخل لایه های مخفی ندارند ، مناسب داده های دنباله ای یا Sequential نیستند و فقط ورودی فعلی را در نظر میگیرند و امکان بخاطر سپردن داده ها و ورودی های قبلی را ندارند.

برخی از داده های دنباله ای یا Sequential

- متن ها : دنباله ای مرتبط از کلمات
- سری های زمانی : داده هایی نظیر داده های مالی در طول زمان و یا چنین مسائلی

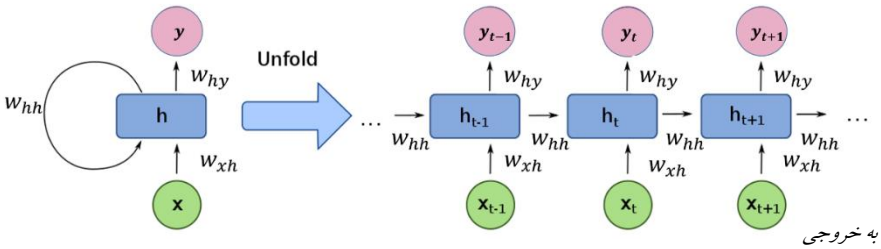


ساختار شبکه های عصبی برگشتی



1:1	1:N	N:1	N:N	N:N
شبکه های معمولی	شبکه های برگشتی	شبکه های برگشتی	شبکه های برگشتی	شبکه های برگشتی
مثال : یک تصویر وارد میکنیم و کلاس بندی میکنیم.	مثال : Image Captioning	مثال : Text Classification Sentiment Analysis	مثال : ترجمه ماشینی	مثال : پردازش و کلاس بندی تک تک فریم های ویدیو

شبکه های بازگشتی دارای state داخلی هستند که در هر مرحله برای تصمیم گیری به آن state نیز نگاه میکنند ، اگر RNN را باز کنیم (آنفولد کنیم) در بازه های زمانی و time step های مختلف به شمای زیر میرسیم.



x_t: ورودی در لحظه t/ام

y_t: خروجی در لحظه t/ام

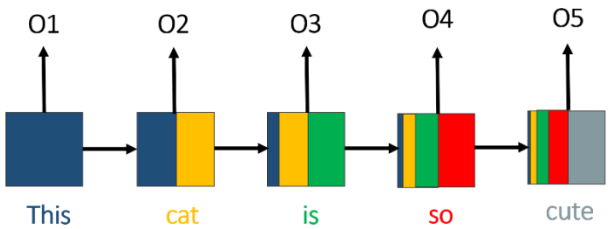
h_t: مقدار state در لحظه t/ام

W_{xh}: وزن های X به ورودی به Y به مخفی

W_{hh}: وزن های Y به مخفی یا state

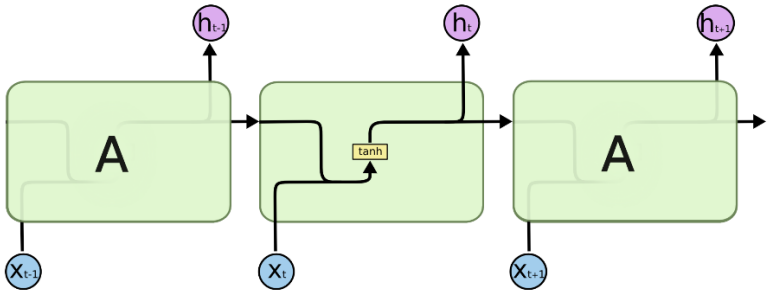
W_{hy}: وزن های Y به خروجی یعنی از Y به مخفی به خروجی

بعنوان مثال ، همانطور که مشاهده میکنید در تصویر زیر ، یک جمله را بطور مثال به شبکه RNN دادیم و شبکه در هر time step یا یک state زمانی ، علاوه بر در نظر گرفتن ورودی state فعلی به ورودی های گذشته نیز attention دارد و سمانتیک جمله را بصورت ترکیبی از یک دنباله از کلمه فعلی و همه کلمات قبل آن تا کلمه اول متن در نظر میگیرد و سمانتیک و مفهوم هر کلمه در این دنباله در مفهوم نهایی برداشت شده توسط مدل تا این state اثر گذار است. و در نهایت خروجی نهایی شبکه یعنی o5 خروجی کل جمله است و میتوانیم آن را به یک شبکه feed forward دهیم و positive یا negative بودن مفهوم آن را تشخیص دهیم.

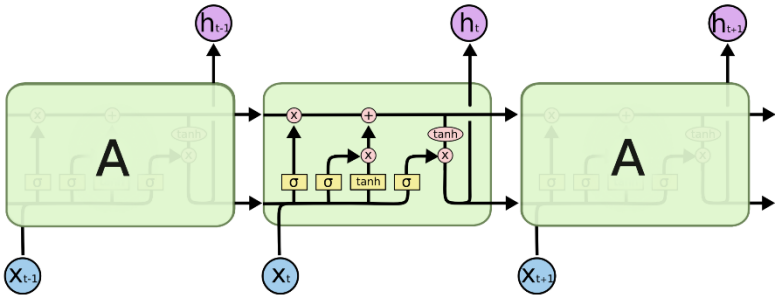


۲-۲ شبکه های LSTM (Long short-term memory network)[10] [7]

دیدیم که یکی از مهمترین مشکلات شبکه های بازگشتی مشکل long term dependency یا وابستگی های بلند مدت بود که با چالش vanishing gradient بوجود می‌آمد. برای حل این مشکل از روش LSTM یا حافظه کوتاه مدت طولانی استفاده میکنیم که از معروف ترین مدل های مبتنی بر شبکه های بازگشتی در پردازش زبان طبیعی محسوب میشود.

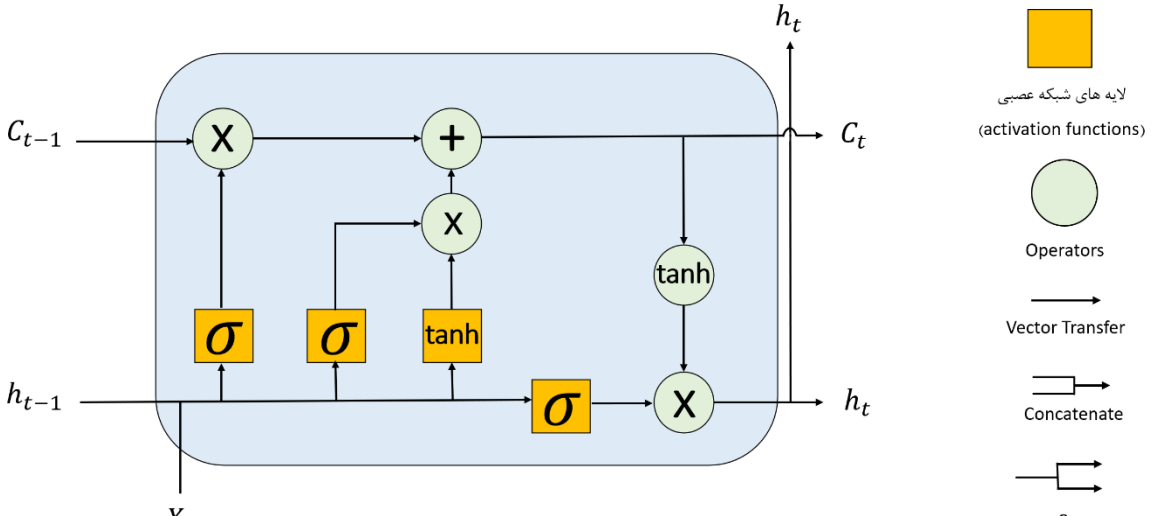


همه شبکه های مبتنی بر RNN ها به شکل زنجیره ای از تکرار ماژول های شبکه های عصبی هستند که در RNN های استاندارد نظیر شکل مقابل ، این ماژول تکرار شونده یک ساختار بسیار ساده مانند یک لایه tanh دارد که جزئیات آن را دیدیم.



شبکه های LSTM هم نوعی از شبکه های RNN هستند که میتوانند با تغییراتی که در ساختار آن ها داده شده است ، وابستگی های طولانی مدت را نیز فرا بگیرند و اطلاعات قدیمی تر را فراموش نمیکنند (حدود چندصد time step).

LSTM ها نیز دارای ساختاری زنجیره ای هستند اما ماژول تکرار شونده LSTM ها به جای داشتن یک لایه عصبی tanh ساده ، دارای چهار لایه تعاملی است که با هم ارتباط برقرار میکنند و سبب کنترل جریان در متن میشوند و میتوانند با این ساختار وابستگی های مهم را در متن به خاطر بسپارند و حفظ کنند و اطلاعات غیر مفید را دور بریزند.

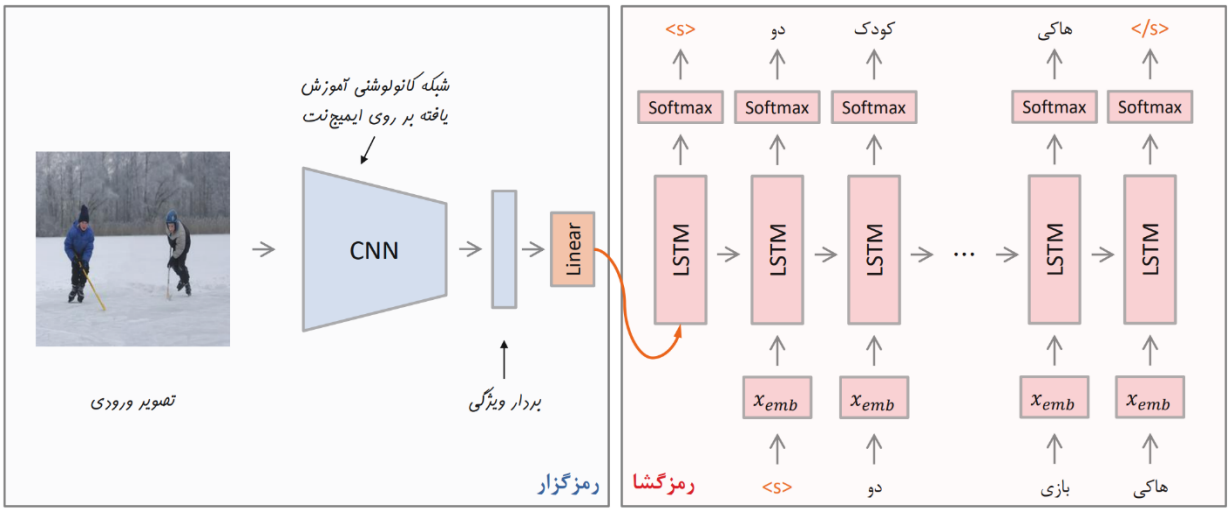


مدل های seq2seq (Decoder-Encoder) : مدل های دنباله به دنباله یا رمزگذار رمزگشا ، معماری متداولی در ترجمه ماشینی است که این معماری را در هرجایی که قرار است یک دنباله ای به دنباله ای دیگر تبدیل شود نظیر ترجمه ماشینی ، بازشناسی صوت ، تشخیص عمل در ویدیو و توصیف تصویر (تبدیل تصویر به متن) و غیره ، میتوانیم استفاده کنیم.

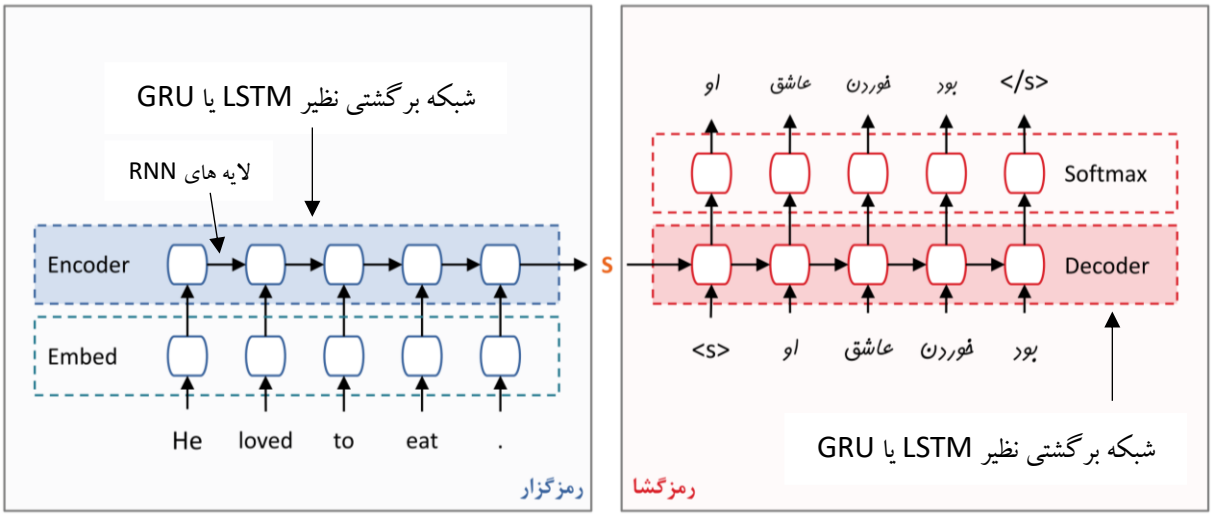
۳-۲ معماری Decoder-Encoder در Image captioning:

در مساله توصیف و کپشن گذاری تصویر مشابه ترجمه ماشینی یک دنباله (البته با یک جز یعنی تصویر) داریم که قرار است به یک دنباله (متن) تبدیل شود.

- ✓ Encoder: تصویر ورودی را دریافت میکند و بازنمایی داخلی از تصویر ایجاد میکند و درنهایت با یک بردار embedding تصویر را پرزنت میکند.
- ✓ Decoder: بازنمایی داخلی یا بردار embedding تصویر را دریافت میکند با رمزگشایی توسط یک شبکه LSTM کلماتی برای توصیف تصویر میسازد.



معماری seq2seq در ترجمه ماشینی : مشابه image captioning با این تفاوت که در اینجا ابتدا دنباله توکن های جمله زبان مبدا را به یک شبکه encoder میدهیم و بازنمایی داخلی آن و بردار embedding آن را دریافت میکنیم و این بردار embedding را به شبکه decoder میدهیم و ترجمه توکن به توکن را براساس این بردار انجام میدهد و دنباله توکن های ترجمه شده زبان مقصد را تولید میکند.



در سمت decoder در مساله توصیف تصویر یا ترجمه ماشینی قرار است یک جمله تولید شود بنابراین کلمات ابتدا به بردار های embedding تبدیل میشوند و درنهایت کار لایه مخفی در decoder یعنی h برای هر کلمه یک بردار احتمال با تابع Softmax تولید میکند که بیشترین احتمال مربوط به ایندکس کلمه ای در corpus است که شبکه بعنوان ترجمه کلمه از زبان مبدا پیش بینی میکند. یا در بحث توصیف تصویر بعنوان کلمات توصیف شده حاصل از توجه به بخش های مختلف تصویر پیش بینی و تولید میکند.

مکانیزم توجه در یادگیری decoder [9]:

محاسبه وزن های هر بخش از تصویر : در ترجمه ماشینی هر کلمه همانطور که گفتیم باید به قسمت های مختلف کلمه مبدا توجه کنیم ، محاسبه وزن و اهمیت کلمات براساس شبهات کسینوسی و Dot product بردار ها میتوانیم استفاده کنیم. مقدار مجموع این وزن ها هم برابر یک میشود. عملا با مکانیزم توجه به خلاصه تمام state های جمله مبدا و ضرایبشان برای ترجمه هر کدام از کلمات توجه میکنیم. محاسبه این ضرایب توسط یک شبکه خیلی ساده میتواند آموزش داده شود ، در نهایت تمام این بردار های وزن دار کلمات بصورت چکیده شده با هم جمع میشوند و به شبکه LSTM در Decoder داده میشود ، مکانیزم توجه بهبود خیلی زیادی در ترجمه ایجاد میکند و علاوه بر ترجمه ماشینی در فیلد های دیگری نظیر image captioning و بحث مورد نظر ما وفیلد ها دیگر نیز میتوانیم از آن استفاده کنیم بطور مثال در تصویر زیر مشاهده میکنید که برای توصیف یک تصویر میتوانیم در هر بار به بخش های مختلف آن تصویر توجه کنیم و بسته به آن بخش تصویر را کپشن گذاری و توصیف کنیم.

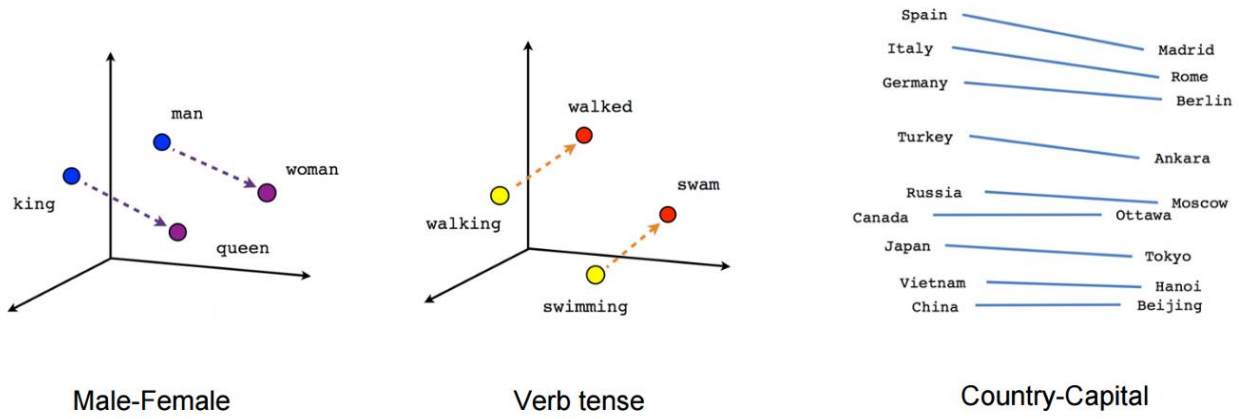
https://distill.pub/2016/augmented-rnns/


A woman is throwing a frisbee in a park.


A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

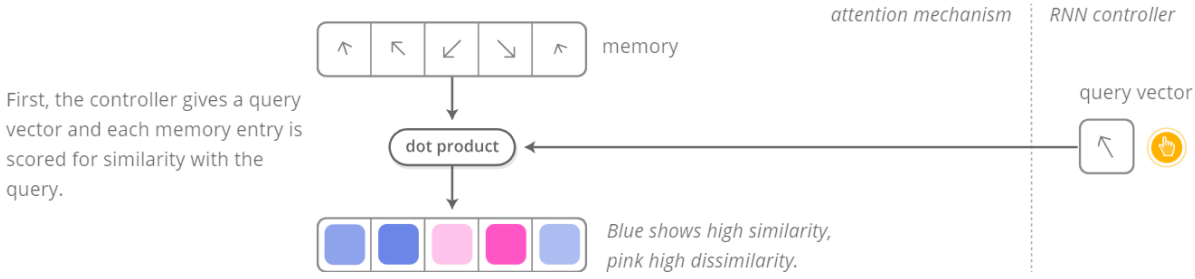
word embedding و بردار سازی کلمات: همانطور که میدانیم برای اینکه مدل و شبکه عصبی یک کلمه را دریافت کند و محاسباتی را روی آن انجام دهد و مثلا در بحث مکانیزم توجه این کلمات را بتواند با یکدیگر از لحاظ سمانتیک مقایسه کند باید این کلمات را تبدیل به عدد کنیم و آن ها را به فضای vector space ببریم ، در گذشته تکنیک های مختلفی برای این کار وجود داشت نظیر bag of word و TF-IDF که مشکل این بردار ها اسپارس بودن آن هاست و مشکلات حافظه را برای ما ایجاد میکنند و علاوه بر این بردار های مفهومی را برایمان تولید نمیکنند و صرفا براساس تکرار کلمات ساخته شده اند که مفهوم خیلی مناسبی و قدرتمندی برای ارائه سمانتیک یک کلمه نیست ، برای اینکه مفهوم کلمات را بصورت مناسب دریافت کنیم به شبکه های عصبی مهاجرت میکنیم و مثلا برای هر کلمه یک بردار خواهیم داشت مثلا با طول 256 ، میتوانیم این بردار ها را با تکنیک های کاهش بعدی نظیر PCA به دو بعد خلاصه کنیم و بصری سازی انجام دهیم ، میبینیم که ارتباط معنا دار و نزدیکی معنا داری بین کلمات نزدیک به هم از لحاظ معنایی پیدا میشود.



Male-Female Verb tense Country-Capital

https://towardsdatascience.com/creating-word-embeddings-coding-the-word2vec-algorithm-in-python-using-deep-learning-b337d0ba17a8

همانطور که مفهوم بردار سازی کلمات یا Word2Vec را دیدیم ، در مکانیزم توجه میتوانیم از این مفهوم استفاده کنیم بطور مثال در تصویر زیر ، یک بردار کوئری با ۶ بردار مقایسه شده است ، مقایسه با dot product یا شباهت کسینوسی صورت میگیرد (بردارها برای بصری سازی صرفا فلش در نظر گرفته شده اند در مساله ترجمه واقعی مثلا میتوانند خروجی یک شبکه برداری به طول 300 یا اندازه های دیگر باشد) ، شباهت بردار کوئری را با هر یک از بردار ها مشاهده میکنید ، بردار های شبیه تر از لحاظ معنایی ، رنگ آبی پررنگ دارند.



بخش سوم : ترنسفورمر ها و بینایی ماشین [1]

در این بخش قصد داریم مکانیزم و معماری ترنسفورمرها را که اولین بار در مقاله به شدت مهم و تاثیر گذار گوگل روی فیلد NLP در سال ۲۰۱۷ با عنوان Attention is all you need مطرح شد ، بررسی کنیم این مقاله در ابتدا در فیلد NMT (Neural machine translation) مورد توجه بود و سپس گوگل در مقاله Vision Transformer از ایده آن در بحث بینایی ماشین استفاده کرد و در این پژوهش نیز از همین رویکرد برای توصیف تصاویر استفاده میشود. ابتدای کار مشکلات شبکه های بازگشتی را که در فیلد ترجمه ماشینی با آن مواجه بودیم مشاهده خواهیم کرد که ترنسفورمر ها چگونه ایده کار را از شبکه های RNN عوض کردند و سبب تحول بزرگی در این فیلد شدند.

چالش های مهم شبکه های بازگشتی

- Long term dependency: در ترجمه ماشینی زمانی که از شبکه های بازگشتی استفاده میکنیم ،کل اطلاعات جمله زبان مبدا را در یک بردار توسط انکودر تولید میکنیم و این کار سبب میشود اگر جمله ای خیلی طولانی شود ، اطلاعات کلمات اول جمله را از دست میدهیم ، که برای حل این مشکل دیدیم که LSTM ها با گیت هایی که داشتند information flow را کنترل میکنند ولی باز هم اگر کلمات طولانی شوند جمله در بردار حاصل encoder ممکن است به خوبی خلاصه نشود و بیشتر تمرکز به روی کلمات آخر بیافتد. فرض کنید چند صد کلمه داشتیم و قرار بود این چندصد کلمه تبدیل میشد به یک بردار و به دیکدر داده میشد و این کار به شدت accuracy را پایین میبرد. برای حل این مشکل هم از مکانیزم attention هم استفاده میکردیم. یعنی بجای اینکه فقط از state نهایی استفاده کنیم ، از تمام state های encoder با جستجو استفاده میکردیم. ولی باز هم اطلاعاتی از دست میرود ، فرض کنید در زمان ترجمه کلمه چند صدم به کلمه اول جمله اشاره کند ، اطلاعات در داخل انکدر هرکاری کنیم باز هم از بین میرود و بهترین حالت این است کد در خود معماری encode و decoder مکانیزم توجه داشته باشیم که خواهیم دید ترنسفورمر ها این مزیت را ایجاد میکنند.
- چالش پردازش sequential در RNN: ساختار شبکه های بازگشتی قابل موازی سازی نیست.
- محو و انفجار گرادیان : حتی با وجود معماری هایی نظیر LSTM و GRU هم محو گرادیان برای دنباله های طولانی خواهیم داشت (البته ترنسفورمر ها هم چالش هایی نظیر نیاز به دیتای زیاد دارند).

معماری Transformer [1]

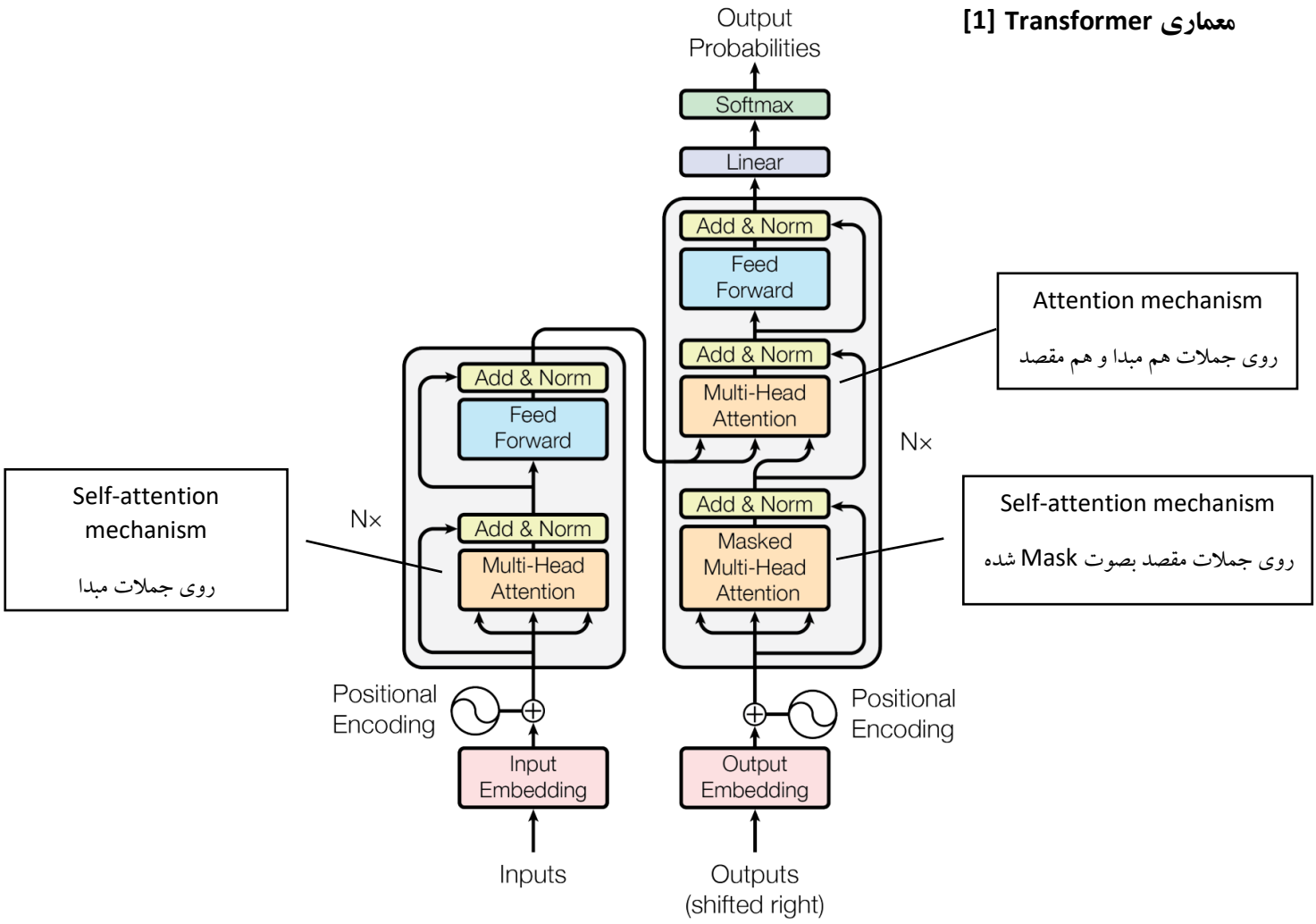


Figure 1: The Transformer - model architecture.

معماری Encoder در Transformer ها [1] **:** کلمات زبان مرجع وارد این بخش میشود ، تمام کلمات در ابتدا تبدیل به word embedding هایشان میشوند سپس وارد input embedding میشوند ، این ورودی ها بصورت یک ماتریس وارد شبکه میشوند مثلا فرض کنید دنباله ورودی طول ۴ کلمه دارد و word embedding ها هم طول ۵۱۲ دارند ، یک ماتریس با ابعاد (4, 512) وارد شبکه میشود و در نهایت خروجی آن مفهوم جملات درون جمله مبدا خواهد بود.

Positional encoding**:** این بخش هم یک ماتریس است که اطلاعات موقعیت کلمات در جمله را دخیره میکند هدف این است که نیاز داریم اطلاعات مربوط به position کلمات را در دنباله داشته باشیم ، البته این بخش در RNN ها نبود چون RNN ها به خودی خود ساختاری بازگشتی و همگی به هم متصل داشتند نیازی به این اطلاعات نداشتند.

Multi head attention: همان مکانیزم توجه است که در داخل خود encoder یا decoder پیاده سازی میشود یعنی ارتباطات معنایی بین کلمات در جمله را بهتر میتوانیم کد کنیم. ماتریس اولیه یعنی نتیجه حاصل از بردار ورودی به input embedding بعد از عبور از positional embedding با multi head attention جمع میشود.

Feed forward: یک شبکه MLP با یک لایه مخفی که در نهایت نتیجه ای را حاصل میدهد ، این نتیجه حاصل همانظور که در شکل میبینید (نوشته شده است N x) دوباره وارد encoder میشود تا اطلاعات بهتری از جمله استخراج شود ، مشابه انسان که یک متن را یکبار میخواند مقداری را متوجه میشود ، بار دیگر میخواند به بخش دیگری توجه میکند و الی آخر و توصیف تصویر هم دقیقا به همین شکل هر بار انسان به بخش خاصی از تصویر توجه میکند.

معماری Decoder در Transformer ها [1]

Masked multi head attention: تفاوت این مکانیزم توجه با مکانیزم توجه در انکدر در این است که در اینجا فقط به کلمات تولید شده مقصد تا مرحله فعلی توجه میکنیم ، طبیعی است چون بقیه کلمات هنوز تولید نشده اند.

نگاه دقیق تر به مکانیزم Multi head attention [1]: همانطور که در مقاله مشاهده میکنیم ، multi head attention یکسری scaled dot product attention است که با هم concatenate شده اند ، بنابراین کل ساختار کار همین scale dot product attention است که در آن سه بخش Query، Key و Value داریم که در ادامه بررسی میکنیم.

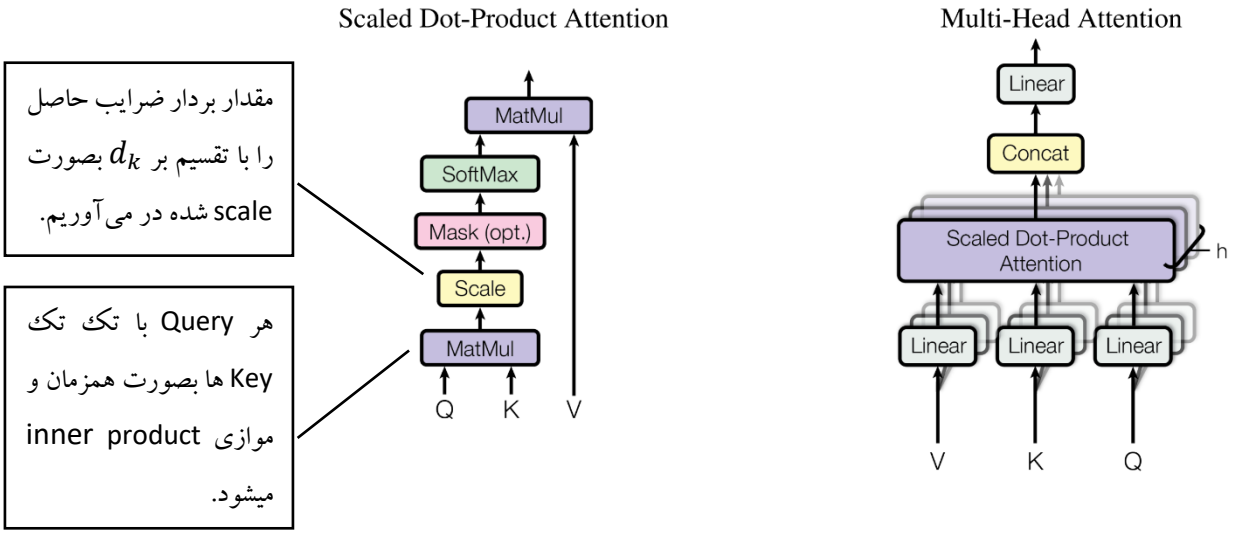


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

در scale dot production ، بردار Q در تک تک بردار های K ضرب یا dot product میشود و نتیجه همانطور که میدانیم شباهت هر بردار Key با بردار Query خواهد بود ، سپس تحت یک تابع Softmax برای اینکه بردار ضرایب و اهمیت هر کدام از Key ها را نسبت به Query بدست آوریم ، به یک بردار تبدیل میشود که مجموع عناصر آن یک میشود و هر عدد نشان دهنده اهمیت آن عنصر خواهد بود که چقدر توجه بگیرد (مشابه مکانیزم توجه که در بخش های قبلی معرفی کردیم). کاملا مشخص است که برای ضرب کردن باید سایز Q و K ها باید برابر باشد و به ازای هر K یه Value داشته باشیم.

$$\text{Attention}(Q,K,V)=\text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

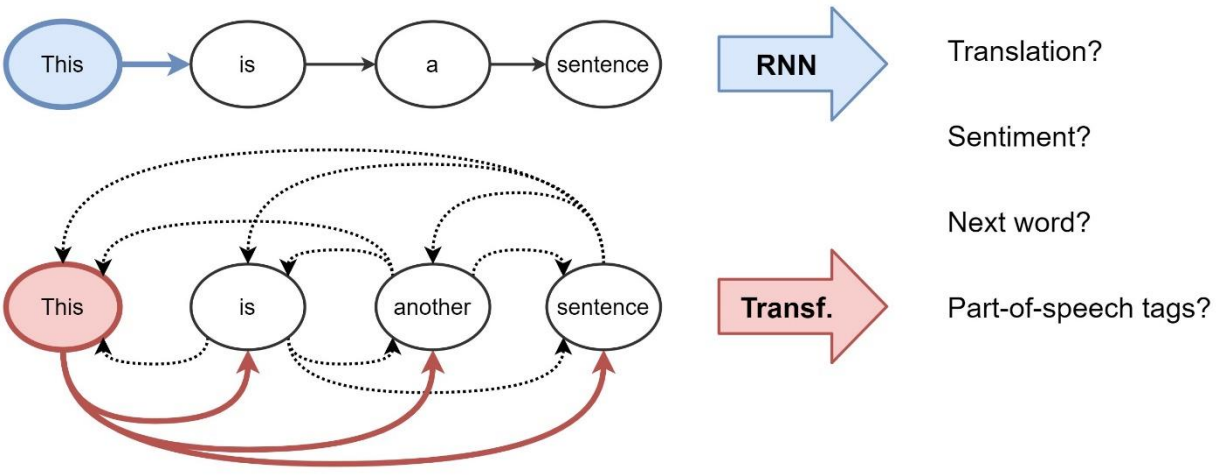
d_k: keys of dimension

- ✓ Mask در معماری Scale dot product attention ، یک ماتریس هم اندازه با ماتریس حاصل است که جاهایی که نمیخواهیم اطلاعات توسط بردار گرفته شود را در این ماتریس برابر با منفی بینهایت قرار میدهیم ، مثلا در سمت دیکدر نمیخواهیم که کلمه مورد نظر از کلمات بعدی اش اطلاعات بگیرد.
- ✓ بردار های Query ، Key و Value: اصلی ترین پارامتر هایی که شبکه میخواهد یاد بگیرد این پارامتر ها هستند ، هر کدام از کلمات مبدا یک بردار Q یک بردار K و یک بردار V تولید و Generate میکنند و هر کدام از کلمات در کل جمله جستجو میکند و اطلاعات کلیدی جمله و کلمات نسبت به هم محاسبه میشود.

نگاه دقیق‌تر به Positional encoding [1]: همانطور که پیش از این گفتیم برخلاف شبکه های بازگشتی به خودی خود دنباله کلمات را دارند در ترنسفورمر ها باید اطلاعات position کلمات را نیز به word embedding اضافه کنیم. فرمولی که برای اینکار در مقاله معرفی شده است براساس تابع های سینوس و کسینوس کار میکند.

$$PE_{(pos,2i)}=sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)}=cos(pos/10000^{2i/d_{\text{model}}})$$

بنابراین در ترنسفورمر ها برخلاف شبکه های بازگشتی درون جمله مبدا و مقصد نیز مکانیزم attention داریم مشابه مثال زیر ، این مزیت باعث میشود مفهوم یک دنباله با بازنمایی کامل داخلی عناصر آن به بهترین شکل دریافت شود.



ارزیابی و نتایج ترنسفورمر ها روی دیتاست ترجمه ماشینی آلمانی به انگلیسی [1]

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	**$3.3 \cdot 10^{18}$**	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

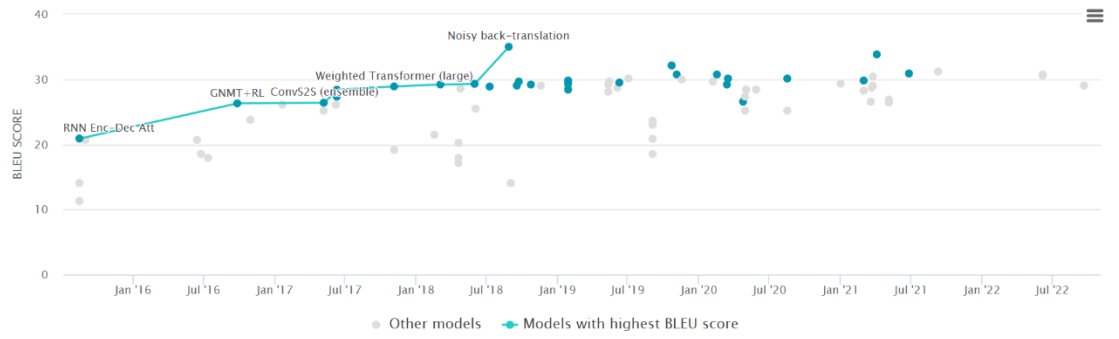
Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

همانطور که مشاهده میکنیم ، رتبه های اول مدل ها همگی مدل های ترنسفورمی هستند.

Machine Translation on WMT2014 English-German

Leaderboard Dataset

View BLEU score by Date for All models



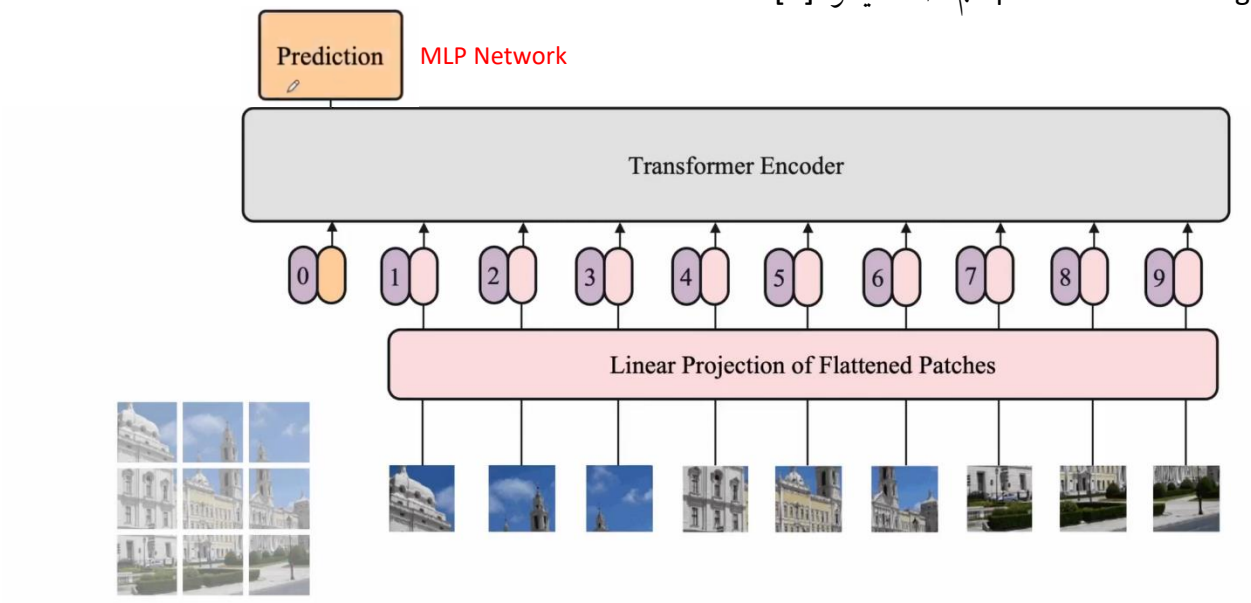
Filter: Transformer LSTM CNN multiscale untagged Hardware Burden Operations per network pass

Edit Leaderboard

Rank	Model	BLEU score	SacreBLEU	Number of Params	Extra Training Data	Paper	Code	Result	Year	Tags
1	**Noisy back-translation**	35.0	33.8		✓	Understanding Back-Translation at Scale			2018	
2	**Transformer+Rep** (Uni)	33.89	32.35		✓	Rethinking Perturbations in Encoder-Decoders for Fast Training			2021	Transformer
3	**T5-11B**	32.1		11110M	✓	Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer			2019	Transformer

ترنسفورمر ها در بینایی ماشین (Vision Transformer) [2]

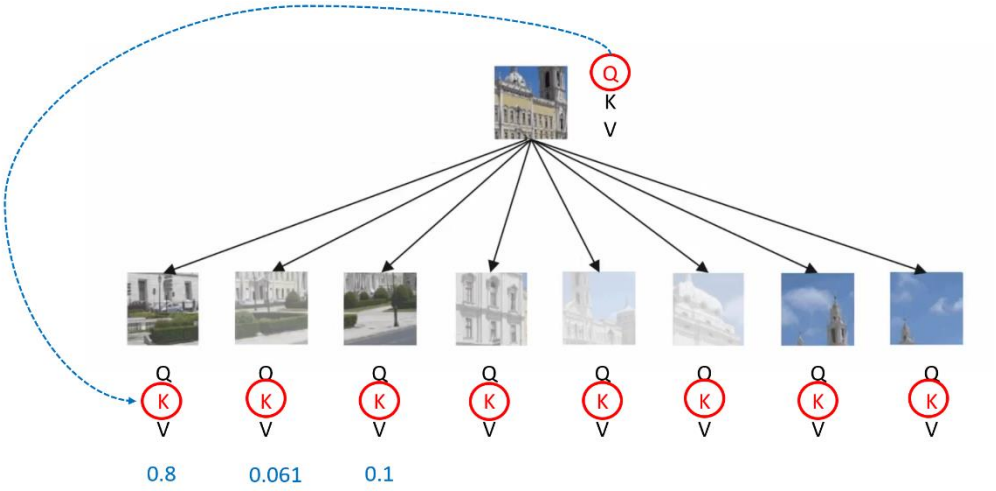
در مقاله An Image is Worth 16x16 Words Transformers for Image Recognition at Scale در سال ۲۰۲۱ گوگل از ایده ترنسفورمر ها که پیش از این در سال ۲۰۱۷ ارائه کرد و با استقبال و اثر گذاری زیادی روی فیلد پردازش زبان طبیعی و مدل های ترجمه ماشینی همراه بود ، استفاده کرد و این ایده و مکانیزم توجه را به فضای بینایی ماشین آورد. اما دلیل مهاجرت از شبکه های کانولوشنی به مکانیزم ترنسفورمر چه بود؟ همانطور که میدانیم عملیات کانولوشن در هر مرحله بخشی از تصویر را میبیند و فیلتر را روی آن اعمال میکند و ما در طول این فرآیند receptive field global و دانش کلی و دید کلی نسبت به عکس نداریم و عملا فقط نسبت به یک ناحیه یعنی ناحیه پنجره کانولوشن نسبت به تصویر اطلاعات دارند و هدف مکانیزم attention و ترنسفورمر این است که تمام قسمت های عکس نسبت به هم اطلاعات بگیرند به همین دلیل تصویر را تبدیل میکند به یک سری non overlapping patch که در تصویر مشخص است. که این patch ها مشابه یک پنجره کانولوشن است که window size آن با stride آن برابر است. هر کدام از patch تبدیل به یک بردار میشود و دقیقا کل این عملیات مشابه عملیات کانولوشن است که تعداد فیلتر های ما هم میشود ابعاد این بردار های حاصل که آن ها را flatten میکنیم. با داشتن این بردار های flat شده میتوانیم این بردار هارا به عنوان token های ورودی یک ترنسفورمر استفاده شوند و به آن ها در ادامه یک positional encoding هم اضافه میشود[2].



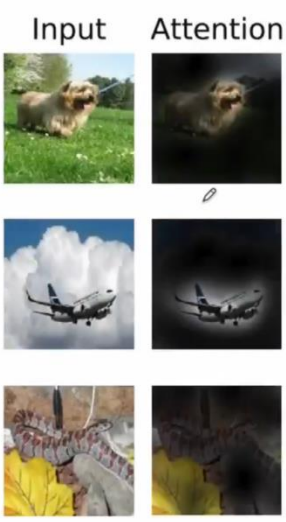
در تصویر بالا ۹ تا patch از عکس داریم که هر کدام یک بردار دارد و عملا ۹ تا بردار هم داریم با ابعاد d بنابراین (9xd) ابعاد ورودی به encoder ترنسفورمر میشود و البته با ابعاد (9xd) از positional encoding ها جمع میشود.

خروجی encoder ترنسفورمر مشابه ورودی همان است و عملا (9xd) تا بردار خروجی از encoder داریم. هر کدام از این توکن های خروجی حاصل توجه patch مربوط به آن به تمام patch های تصویر بوده و عملا در این روش دید کلی و receptive field global داریم و هر patch عملا میتواند معنی خود را در عکس پیدا کند.

برای تسک های مختلف میتوانیم از این اطلاعات معنادار حاصل یعنی بردار های خروجی حاصل استفاده کنیم مثلا برای تسک image classification میتوانیم تمام آن ۹ بردار را average pooling بگیریم و به یک بردار حاصل برسیم که بعدا آن بردار را به MLP میدهیم و از آن برای classification استفاده میکنیم[2].



هر کدام از patch ها باید یادبگیرد که براساس اطلاعات داخلش یک بردار Query بسازد که بدنبال آن در تصویر میگردد و هر کدام از patch های دیگر نیز باید براساس اطلاعات خود نیز باید یاد بگیرند که یک Key درست کنند که این یک لایه Dense بصورت Learnable است. Query با استفاده از عملیات شباهت کسینوسی شباهت خود را با بقیه patch های تصویر میسنجد و در نهایت به آن ها امتیازی با SoftMax میدهد که جمع همه با هم یک میشود. این key ها مشابه headline یک داکیومنت در بحث بازیابی اطلاعات هستند که شباهت آن ها با query مورد نظر ما تعیین میکند که چقدر قصد داریم از آن اطلاعات بگیریم و ضریب بدست آمده در Value ها ضرب شده و در نهایت همه آن ها با هم جمع میشوند و در نهایت اطلاعاتی که این patch از بقیه تصویر میتواند بگیرد حاصل میشود. هدف نهایی این است که بهترین Q و K و V را یاد بگیریم بطوریکه در تصویر روبه رو با ورود یک تصویر در سمت چپ مثلا تصویر یک هواپیما تمام یا بیشتر اطلاعات نهایی حاصل از Average pooling از بخش و ناحیه patch های هواپیما در تصویر بگیرد[2].



بخش چهارم : توصیف تصویر با استفاده از Vision Transformer [2]

همانطور که در تصویر مشاهده میکنیم و پیش از این با مفاهیم ترنسفورمرها و کاربرد آن ها در بینایی ماشین آشنا شدیم میتوانیم بعنوان ورودی بخش انکدر ترنسفورمر تصویر خود را به patch هایی تبدیل کرده و سپس با positional encoding آن جمع نموده و در نهایت بردار های حاصل را وارد یک انکدر ترنسفورمر کنیم که پیش از این عملکردش را بررسی کردیم و در این انکدر با استفاده از مکانیزم توجه هر patch تصویر به patch های دیگر در نهایت بردار های معنا داری خواهیم داشت که هر کدام بخشی از تصویر را بازنمایی میکنند در تسک classification برای رسیدن به یک بردار و دادن آن بردار به شبکه MLP برای تعیین نوع کلاس تصویر، آنهارا Avg Pool میگرفتیم اما در اینجا هر کدام از این بردار هم بیان کننده معنایی از تصویر است که میتواند به دیکدر برای تبدیل متن داده شود.

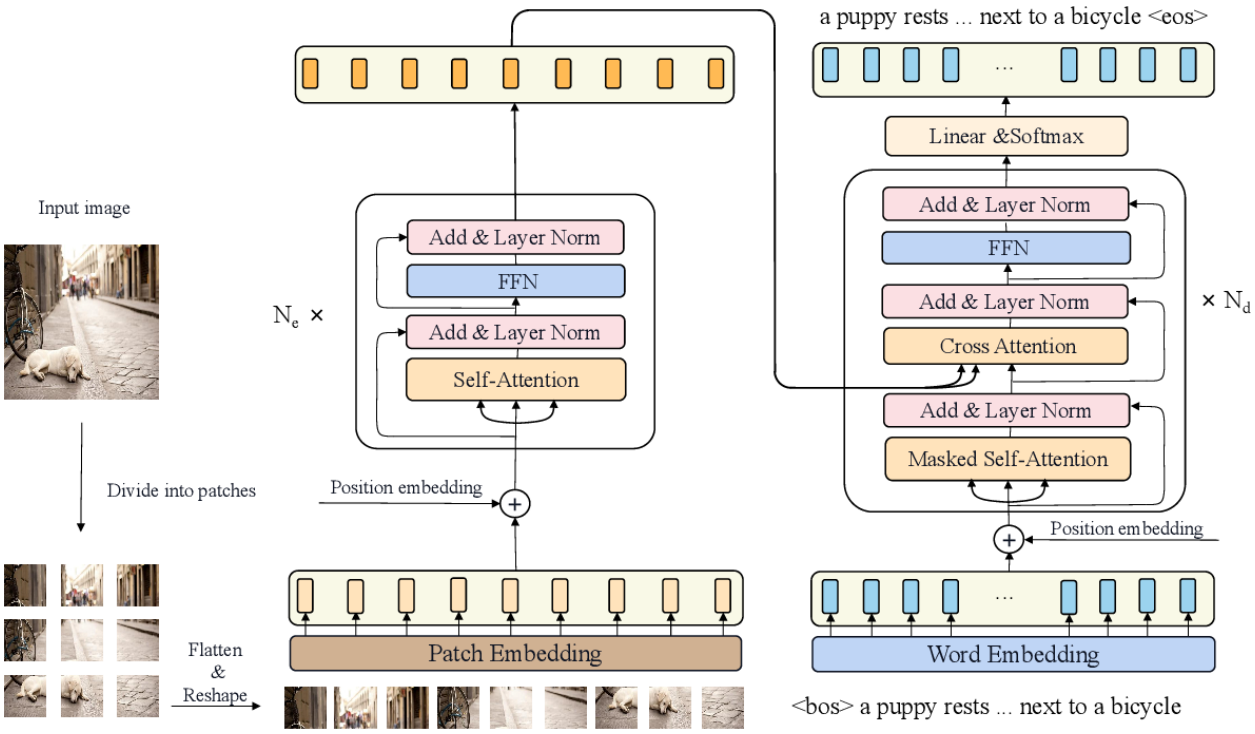


Figure ۳ : Figure 1: CPTR: Full Transformer Network for Image Captioning https://arxiv.org/pdf/2101.10804.pdf

دیکدر در این معماری در هر مرحله وظیفه دارد با گرفتن word embedding قبلی و بردار های حاصل از انکدر نتیجه ای را پیش بینی کند که این مراحل n بار تا تولید توکن انتهای کپشن پیش میرود. برای قرار گیری درست نتایج یا توکن ها یا کپشن نهایی حاصل در کنار هم و یک جمله با قواعد دستوری درست نیاز است که دیکدر براساس یک مدلسازی زبانی عمل کند که عملا این بخش مهم از کار دیکدر در این معماری است که در ادامه با مدل سازی زبانی هم آشنا میشویم.

مدل سازی زبان (Language model) و Vision Transformer

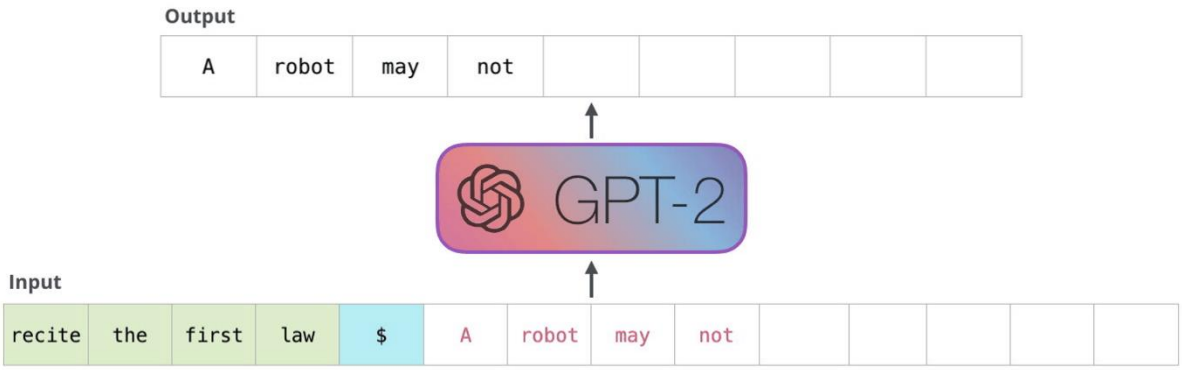
مدل سازی زبان به معنی محاسبه یک تابع توزیع احتمال است و انتساب یک احتمال به یک دنباله ای از کلمات است بطوریکه مجموع احتمالات روی تمام دنباله های ممکن برابر یک باشد و برای چنین عملیات از یک تابع Softmax در زمان پیاده سازی میتوانیم استفاده کنیم.

$$P(w_1, w_2, \dots, w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2) \times \cdots \times P(w_n|w_1, w_2, \dots, w_{n-1})$$

میخواهیم احتمال اینکه این زنجیره کلمات از کلمه اول یعنی w1 تا کلمه آخر این دنباله یعنی w2 با این توالی بیایند را محاسبه کنیم

کاربرد مدلسازی زبان : ساخت مدل زبانی برای یک پیکره یکی از مباحث پرکاربرد در پردازش زبان طبیعی است که کاربرد مهم آن را در ادامه معرفی میکنیم.

- Image Captioning : در ترنسفورمر ها پس از دریافت خروجی Encoder بعنوان ورودی در Decoder کلماتی که از روی توجه به بخش های مختلف تصویر ساخته شده اند باید توسط مدل زبانی با ترتیبی درست در زبان مقصد در کنار هم قرار بگیرند مثلا فرض کنید در یک تصویر کلمات : playing ، guitar ، a و man حاصل از توجه ترنسفورمر به بخش های مختلف تصویر بود حال باید ترتیب قرار گیری این کلمات در کنار هم و ساختن جمله a man playing guitar برساس قواعد زبان انگلیسی و کورپس مورد ساخته شود که در این پژوهش این کار را GPT-2 انجام میدهد [3].



که وظیفه آن دقیقا در بخش Decoder یک مدل ترنسفورمر برای تبدیل کلمات با ساختار و جمله بندی درست است.

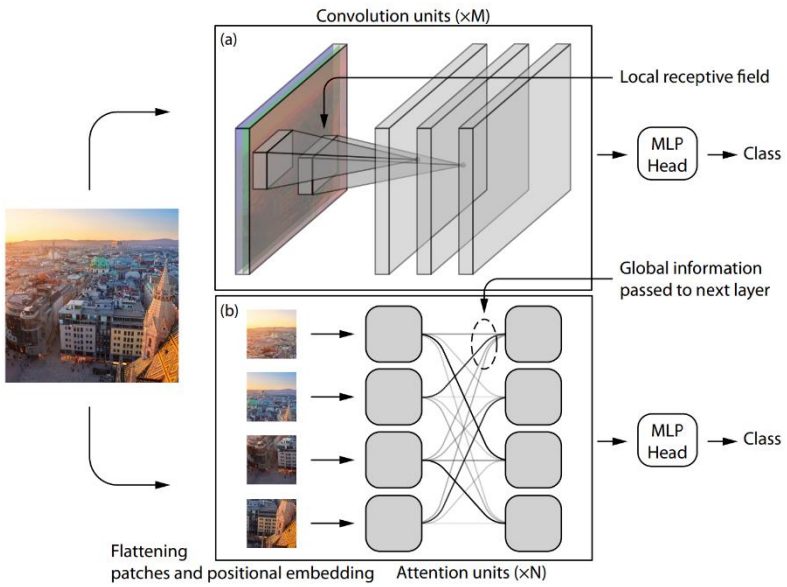
- ترجمه ماشینی : فرض کنید مترجم ماشینی در زمان ترجمه سه کلمه apple ، likes و he را به عنوان خروجی پیش بینی کرده است ولی ترتیب درست قرار گیری این کلمات را در کنار هم نمیداند ، بهترین روش این است که براساس مدل زبانی corpus یعنی زبان انگلیسی احتمال قرار گیری حالت های مختلف این کلمات کنار هم را محاسبه کند و بهترین و پر احتمال ترین دنباله ممکن را بعنوان خروجی مترجم ارائه دهد.
- Speech recognition: انتخاب بهترین کلمه بعدی براساس سیگنال صوتی دریافت شده ، از روی مدل زبانی انتخاب میکنیم که احتمال رخداد کلمات بدست آمده از سیگنال صوتی در پیکره مورد نظر چقدر است.

مدل های زبانی N بخشی مبتنی بر شمارش ساده و احتمالات : محاسبه احتمال رخداد تمام کلمات دنباله در کنار هم همناطور که در فرمول محاسبه توزیع احتمال دیدیم کار زمانبری است و برای محاسبه آن از روش های تخمینی میتوانیم استفاده کنیم ، مثلا بجای در نظر گرفتن رخداد تمام کلمات قبلی میتوانیم در این روش تخمینی احتمال رخداد n-1 کلمه قبلی را تخمین بزنیم که یکی از روش های معروف آن مدل n=2 یعنی ۲ بخشی است که صرفا ۱ کلمه قبلی را نگاه میکند و احتمال رخداد آن را در محاسبه احتمالات به شکل زیر حساب میکند.

مدل دو-بخشی → $P(w_1, w_2, \dots, w_n) \approx P(w_1)P(w_2|w_1)P(w_3|w_2) \times \cdots \times P(w_n|w_{n-1})$

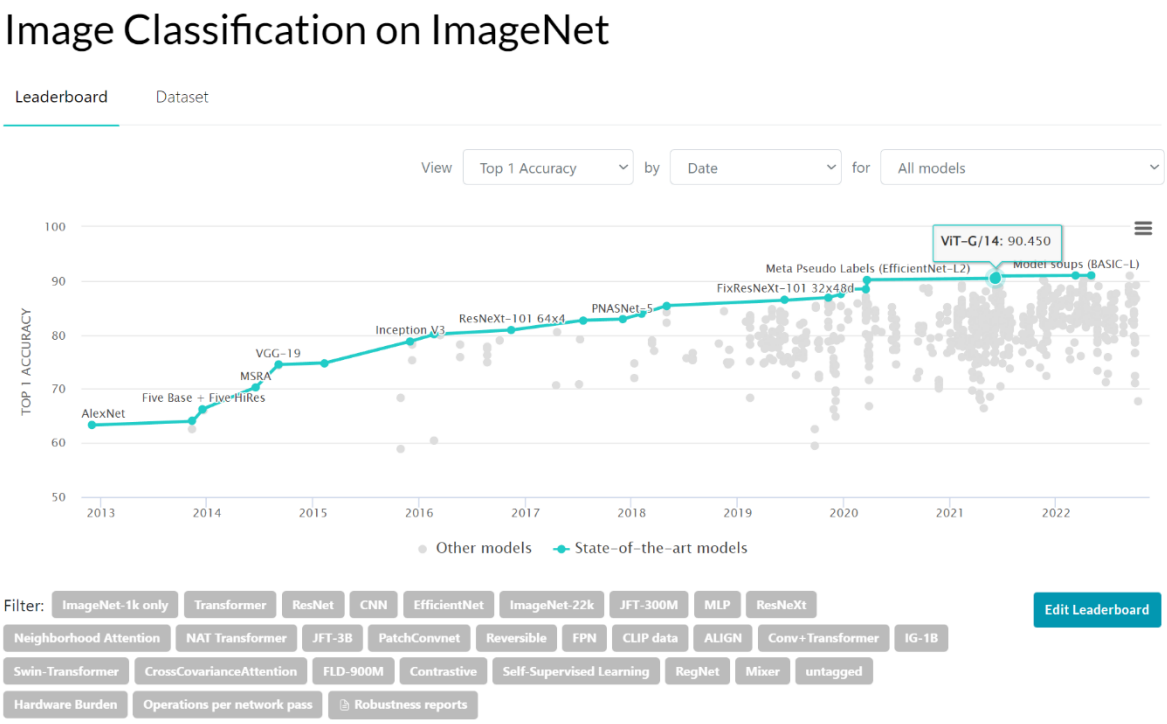
برای تک تک دو کلمه ها حساب میکنیم : احتمال شرطی : $P(w_i|w_{i-1}) = \frac{P(w_{i-1}, w_i)}{P(w_{i-1})} = \frac{N(w_{i-1}, w_i)}{N(w_{i-1})}$

Vision Transformer در مقابل شبکه های CNN: همانطور که در تصویر زیر مشاهده میکنید ، همانطور که پیش از این ویژن ترنسفورمر ها را بررسی کردیم ، دیدیم که برخلاف CNN که local receptive field و دید محلی داشتند معماری ویژن ترنسفورمر توانسته است با توکن سازی patch های تصویر با یک تصویر مشابه یک جمله در NLP برخورد کند و با استفاده از مکانیزم Attention به Global information برسد و این باعث رشد چشم گیر محبوبیت آن پس از معرفی آن در سال ۲۰۲۱ توسط گوگل شده است.



ارزیابی و نتایج Vision Transformer [2]

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	–
CIFAR-100	**94.55** ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	–
Oxford-IIIT Pets	**97.56** ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	–
Oxford Flowers-102	99.68 ± 0.02	**99.74** ± 0.00	99.61 ± 0.02	99.63 ± 0.03	–
VTAB (19 tasks)	**77.63** ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k



بخش پنجم : پیاده سازی Image Captioning در پایتون [12] و نتایج

ابزار ها و زبان برنامه نویسی: کتابخانه Transformers در پایتون و بستر Hugging Face و دیتاست مورد استفاده ی مدل از پیش آموزش داده شده دیتاست COCO است که در آن تصویر مختلف با ۵ کپشن وجود دارد و از لینک زیر بصورت عمومی قابل دسترسی است.

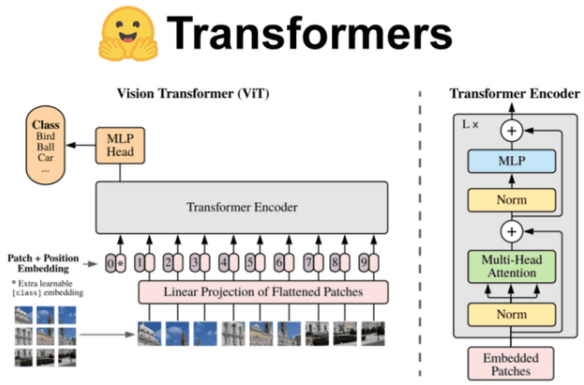
[Microsoft COCO Dataset]

http://cocodataset.org/#captions-2015

More than 80k training images and 40k validation images.

At least 5 captions for every image.

پروژه توصیف تصاویر براساس مدل Vision Encoder Decoder و ViT Feature Extractor با استفاده از کتابخانه transformers روی داده های از پیش آموزش دیده ، پیاده سازی شده است که در ادامه لینک دسترسی به پیاده سازی قرار گرفته است.



https://huggingface.co/ydshieh/vit-gpt2-coco-en-ckpts/tree/main

نتایج پژوهش و پیاده سازی توصیف تصاویر

مدل از پیش آموزش داده شده توسط آقای Yih Dar بر اساس متریک و معیار RougeL به مقادیر زیر دست یافته است.

Epoch... (5/30 | Step: 11535 Eval Loss: 1.8884644508361816 Eval rougeL: 37.3854

Eval rouge1: 41.1904 Eval rouge2: 15.6084

Eval rougeLsum: 37.3791 Eval gen_len: 11.4548

پیاده سازی نهایی Image Captioning در بستر وب [12]

https://github.com/M-Taghizadeh/BigData_Projects/blob/master/Image%20Captioning/Image_Captioning.ipynb

نتایج پژوهش و ارزیابی مدل : در انتهای کار مدل از پیش آموزش داده شده را روی ۱۶۷۸ تصویر از دیتاست COCO Caption 2014 آزمایش و ارزیابی کردیم و متریک مورد استفاده معیار BLEU که یک عدد بین صفر و یک است برای کپشن های داده شده توسط مدل توسط کتابخانه NLTK در پایتون محاسبه شده است. این معیار عددی بین صفر و یک است که میزان شباهت دو بردار جمله را با هم محاسبه میکند و در پژوهش کپشن تولید شده را با جملاتی که برای تصویر انسان ها تولید کرده اند مقایسه کردیم.

```
print(human_caption[200])
print(vit_caption[200])
```

```
['Several books are stacked on a table. ', 'A set of books sitting on top of a shelf.', 'A few books lined up beside each other on the shelf. ', 'The books are stacked on the shelve on the rack ', 'A shelf with many dif ferent types of books.']
a stack of books on top of a shelf
```

```
len(human_caption), len(vit_caption), len(results)
```

```
(1678, 1678, 1678)
```

Calculate BLEU Score using NLTK (sentence_bleu)

```
from nltk.translate.bleu_score import sentence_bleu, SmoothingFunction

sum_blue = 0
for i in range(len(results)):
    reference = []
    for item in human_caption[i]: reference.append(item.split())
    candidate = vit_caption[i].split()
    sum_blue += sentence_bleu(reference, candidate, smoothing_function=SmoothingFunction().method4)
print(f"Average BLEU score -> {sum_blue/len(results)}")
```

```
Average BLEU score -> 0.25589140693836127
```

منابع و مقالات مرجع در پژوهش و توسعه پروژه

[1] A. Vaswani et al.
Attention is All you Need
Advances in Neural Information Processing Systems, 2017

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby.
An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale
9th International Conference on Learning Representations, {ICLR} 2021

[3] García Gilabert, Javier
Image Captioning using pre-trained GPT-2 models
universitat politècnica de valència, 2021

[4] Li, Minghao and Lv, Tengchao and Chen, Jingye and Cui, Lei and Lu, Yijuan and Florencio, Dinei and Zhang, Cha and Li, Zhoujun and Wei, Furu
TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models, 2021

[5] M. Kaur and A. Mohta.
A Review of Deep Learning with Recurrent Neural Network
International Conference on Smart Systems and Inventive Technology (ICSSIT), 2019

[6] S. Hochreiter and J. Schmidhuber.
Long Short-Term Memory
Neural Computation, 1997

[7] F. Karim, S. Majumdar, H. Darabi and S. Chen
LSTM Fully Convolutional Networks for Time Series Classification
IEEE Access, 2018

[8] B. Lindemann, T. Müller, H. Vietz, N. Jazdi, and M. Weyrich.
A survey on long short-term memory networks for time series prediction
Procedia CIRP, 2021

[9] Luong, Thang, Hieu Pham and Christopher D. Manning.
Effective Approaches to Attention-based Neural Machine Translation
EMNLP, 2015

[10] M. Sundermeyer, H. Ney and R. Schlüter.
From Feedforward to Recurrent LSTM Neural Networks for Language Modeling
IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2015

[11] I. Goodfellow et al.
Generative Adversarial Nets
Advances in Neural Information Processing Systems, 2014

[12] https://github.com/M-Taghizadeh/BigData_Projects