Attention Based Neural Machine Translation

شبکه ماشین ترجمه عصبی مبتنی بر ساختار انکدر-دیکدر مرسوم از دو شبکه عصبی بازگشتی استفاده می نماید. شبکه انکدر وظیفه تبدیل ورودی به یک بردار با طول ثابت را دارد و شبکه دیکدر نیز وظیفه تبدیل بردار به توالی خروجی را ایفا می کند. شبکه مبتنی بر attention به دنبال پیش بینی کلمات خروجی بر اساس مقادیر وزن داده شده به هر state پنهانی است که در مرحله انکدر به دست می آید.

در حالت کلی، از شبکه یک خروجی یکتا بدست می آید که در صورتی که کلمه ای اشتباه پیش بینی شده باشد، خروجی اشتباه خواهد بود. برای بهبود این مشکل دو روش beam search و ensemble decoding پیشنهاد می شود که بهبود پایداری پیش بینی های شبکه کمک می نماید. روش beam search به حفظ فرضیه های بهتر در مرحله دیکدر کمک می نماید و متد ensemble decoding مقدار واریانس خروجی را کاهش می دهد. به همین دلیل این دو روش در بهبود کیفیت ترجمه های حاصل کمک می نماید.

شبکه ماشین ترجمه عصبی از دو بخش انکدر و دیکدر تشکیل شده است که هر کدام از آن شبکه عصبی عمیق معادل زیر است:

Encoder: one-layer bi-directional LSTM

Decoder: one-layer uni-directional LSTM

انکدر:

جمله ورودی یا x به یک توالی از بردارهای کلمات one-hot تبدیل می شود. در هر گام زمانی i، بردار جاسازی کلمه ورودی e به صورت زیر محاسبه می شود:

$$\boldsymbol{e}_i^s = \tanh(\boldsymbol{W}_x \boldsymbol{x}_i)$$

در رابطه بالا w ماتریس وزن است که از روی ابعاد کلمات جاسازی و کلمه ورودی بدست می آید. State پنهانی h نیز در بخش انکد از مجموع state پیشرو و state برگشتی محاسبه می شود. فرمول های زیر نحوه محاسبه state پنهانی را نشان می دهد:

$$\overrightarrow{\boldsymbol{h}_i} = \text{LSTM}(\boldsymbol{e}_i^s, \overrightarrow{\boldsymbol{h}_{i-1}})$$

$$\overleftarrow{\boldsymbol{h}_i} = \text{LSTM}(\boldsymbol{e}_i^s, \overleftarrow{\boldsymbol{h}_{i+1}}).$$

دیکدر:

جمله خروجی یا y به صورت توالی از بردارهای کلمه one-hot با طول مشخص بدست می آید. در هر گام زمانی j، state پنهانی h به صورت زیر محاسبه می شود:

$$\boldsymbol{h}_j = \text{LSTM}([\boldsymbol{e}^t_{j-1} : \tilde{\boldsymbol{h}}_{j-1}], \boldsymbol{h}_{j-1})$$

در رابطه با e نشان دهنده بردار جاسازی کلمه نهایی، h^ نشان دهنده state پنهانی attention و h معرف state پنهانی گام پیشین است.

نحوه محاسبه وضعیت پنهانی attention از روی رابطه زیر است که در آن w معرف ماتریس وزن و b مشخص کننده مقدار بایاس است. همچنین بردار متن c نیز از روی مجموع وزن تمامی state پنهانی انکدر محاسبه می شود.

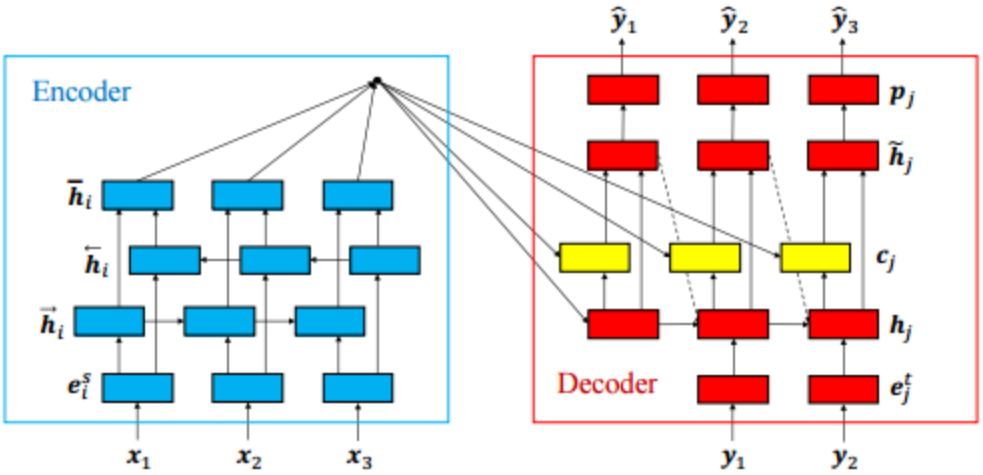
$$\boldsymbol{c}_j = \sum_{i=1}^{|\boldsymbol{X}|} \alpha_{ij} \bar{\boldsymbol{h}}_i.$$

$$\alpha_{ij} = \frac{\exp(\bar{\boldsymbol{h}}_i^{\mathrm{T}} \boldsymbol{h}_j)}{\sum_{k=1}^{|\boldsymbol{X}|} \exp(\bar{\boldsymbol{h}}_k^{\mathrm{T}} \boldsymbol{h}_j)}.$$

مقدار وزن نیز از روی توزیع احتمالات نرمال شده بدست می آید. همچنین احتمال شرطی کلمه خروجی توسط رابطه زیر می شود:

$$\boldsymbol{p}(\hat{\boldsymbol{y}}_j | \boldsymbol{Y}_{<j}, \boldsymbol{X}) = \text{softmax}(\boldsymbol{W}_p \bar{\boldsymbol{h}}_j + \boldsymbol{b}_p)$$

کلمات نادر در محموعه لغات که مخادل مناسب برای آنها وجود ندارد با توکنی مانند <unk> جایگزین می شود. به این صورت زمانی که کلماتی اینچنینی پیش بینی شود، نشانه مانند زیر در خروجی قرار داده می شود. شکل زیر نمایی از دو شبکه encoder-decoder را نشان می دهد.



تابع هدف:

تابع هدف در این مقاله به صورت زیر تعیین می شود که در آن D تعداد داده و تتا پارامترهای مدل است. در این تابع به دنبال بیشینه کردن هدف هستیم. بردار جاسازی مورد استفاده مدل word2vec است.

$$\mathcal{L}(\theta) = \frac{1}{D}\sum_{d=1}^{D}\sum_{j=1}^{|Y|}\log p(y_j^{(d)}|Y_{<j}^{(d)}, X^{(d)}, \theta)$$

Beam search:

در حالت کلی کلمه ای که محتمل تر باشد، به عنوان خروجی انتخاب می شود ولی در این حالت n نمونه محتمل در نظر گرفته می شود. به این صورت ریسک تولید جمله اشتباه کاهش می یابد.

Ensemble Decoding:

در این روش احتمال شرطی کلمه خروجی به صورت میانگین امتیازات محاسبه می شود. M نشان دهنده تعداد مدل ها است. استفاده از این روش ریسک پیش بینی اشتباه در مدل را در هر گام زمانی کاهش می دهد:

$$p(\hat{y}_j|Y_{<j}, X) = \frac{1}{M}\sum_{m=1}^{M} p^{(m)}(\hat{y}_j|Y_{<j}, X)$$

در این مقاله از مجموعه داده ترجمه زبان ژاپنی و انگلیسی استفاده می شود.

به صورت کلی برخی از مهم ترین پارامترهای شبکه عبارتند از:

- Number of hidden units: 1,024
- Word embedding dimensionality: 512
- Source vocabulary size: 100,000
- Target vocabulary size: 30,000
- Minibatch size: 128
- Optimizer: Adagrad
- Initial learning rate: 0.01
- Dropout rate: {0.1, 0.2, 0.3, 0.4, 0.5}
- Beam size: {1, 2, 5, 10, 20}

بعد از تست های مختلف بهترین مقدار نرخ dropout مقدار ۰,۲ بدست آمده است.

در هنگام پیاده سازی یک شبکه مولد خصمانه GAN، شبکه ماشین ترجمه عصبی مبتنی بر attention اشاره شده در این مقاله به عنوان شبکه Generator در مدل GAN ما بکار گرفته می شود. برای اینکه شبکه GAN پایداری بهتری در فرآیند آموزش خود داشته باشد بهتر است تا شبکه های Generator و Discriminator آن پیش آموزش داده شوند. برای پیش آموزش شبکه G از مقادیر وزن حاصل در epoch های مختلف این الگوریتم استفاده می کنیم. به این صورت که بعد از پایان هر دو تمامی وزن ها را نگه داشته و در هنگام پیش آموزش شبکه مولد به جای وزن های تصادفی از این مقادیر بهره خواهیم برد.