مرحله نهایی گزارش پروژه پردازش زبان های طبیعی

یادگیری بردار جملات به روش با ناظر و با استفاده از مجموعه داده استنتاج زبانی استنفورد

توسط:

طاها سماواتی

استاد :

دکتر بهروز مینائی

پاییز ۱۳۹۸


بسم الله الرحمن الرحیم

چـــکیـــده

امروزه بدست آوردن بردار های کلمه و جمله بخش لازم و ضروری هر پروژه پردازش زبان طبیعی مبتنی بر شبکه های عصبی عمیق است. فرایند تبدیل کلمات یا جملات به بردار هایی با طول یکسان توسط روش تعبیه کلمات/جملات[۱] انجام می‌شود.

در سال های اخیر توجه به سمت تعبیه های همگانی[۲] افزایش یافته است. دلیل آن را می‌توان آموزش این مدل ها بر روی مجموعه متن های بزرگ دانست. این مدل ها در بسیاری از کاربرد ها مثل آنالیز احساسات یا ترجمه ماشینی به کار رفته اند و نتایج خوبی به ثبت رسانده اند. این مدل ها بر داده های مورد نظر اعمال شده و بر اساس آموزش قبلی روی داده هایی با حجم بالا به هر کلمه یک بردار با طول ثابت اختصاص می دهند. با وجود پیشرفت ها در زمینه روش های تعبیه کلمات ، هنوز پیدا کردن روشی مناسب برای تولید بردار نماینده برای جملات چالش بر انگیز است.

در مقاله انتخابی روشی با ناظر برای آموزش یک رمزنگار جمله معرفی می شود و سپس عملکرد مدل آموزش دیده روی مجموعه داده های مختلف NLP با وظیفه های مختلف ارزیابی می شود. از مزیت های روش با ناظر بر روش های بدون ناظر پیشین می توان به مواردی چون نیاز به حجم داده کمتر و همچنین زمان آموزش کمتر اشاره کرد. با وجود نیاز کمتر به زمان و داده، عملکرد مدل آموزش دیده به روش با ناظر در بسیاری از وظیفه ها بهتر است.

در این پژوهش به بررسی کامل مقاله انتخابی و همچنین بیان مزیت ها و مشکلات آن پرداخته می شود. همچنین، همه جوانب استفاده از این روش برای استخراج بردار جملات بررسی می شود. در فصل اول مقدمه بیان شده و کارهای پیشین مورد بررسی قرار می گیرد. در فصل دوم روش ارائه شده برای استخراج جملات به روش با ناظر توضیح داده می شود و همچنین مجموعه داده استفاده شده معرفی می‌شود. در فصل سوم نتایج بدست آمده، نمایش داده شده و تحلیل می شوند و در قسمت پایانی جمع بندی و نظرات بیان می شود.

کلید واژه : پردازش زبان طبیعی، روش با ناظر استخراج بردار جملات، یادگیری عمیق، شبکه های کدگذار جمله

---

[۱] Word/Sentence Embedding

[۲] Universal Embeddings

فهرست مطالب

عنوان **صفحه**

فهرست شکل‌ها

عنوان **صفحه**

فهرست جداول

عنوان **صفحه**

فصل ۱- مقدمه و کار های پیشین

۱-۱- مقدمه

امروزه بدست آوردن بردار های کلمه و جمله بخش لازم و ضروری هر پروژه پردازش زبان طبیعی مبتنی بر شبکه های عصبی عمیق است. فرایند تبدیل کلمات یا جملات به بردار هایی با طول یکسان توسط تعبیه کلمات/جملات انجام می‌شود.

در سال های اخیر توجه به سمت تعبیه های همگانی افزایش یافته است. دلیل آن را می توان آموزش این مدل ها بر روی مجموعه متن های بزرگ دانست. این مدل ها در بسیاری از کاربرد ها مثل آنالیز احساسات یا ترجمه ماشینی به کار رفته اند و نتایج خوبی به ثبت رسانده اند. این مدل ها بر داده های مورد نظر اعمال شده و بر اساس آموزش قبلی روی داده هایی با حجم بالا به هر کلمه یک بردار با طول ثابت اختصاص می دهند.

برای تولید تعبیه جملات چندین روش وجود دارد. ساده ترین آنها میانگین گرفتن بردارهای حاصل از تعبیه کلمات برای هر جمله است. روش های دیگری از جمله با ناظر و بدون ناظر وجود دارد. در مقاله انتخابی که توسط تیم تحقیقاتی Facebook ارائه شده است، روشی با ناظر برای ایجاد تعبیه جملات معرفی شده است که از داده های Stanford Natural Language Inference استفاده می کند و نشان می دهد این روش از روش های بدون ناظر مثل Skip thoughts بهتر عمل می کند.

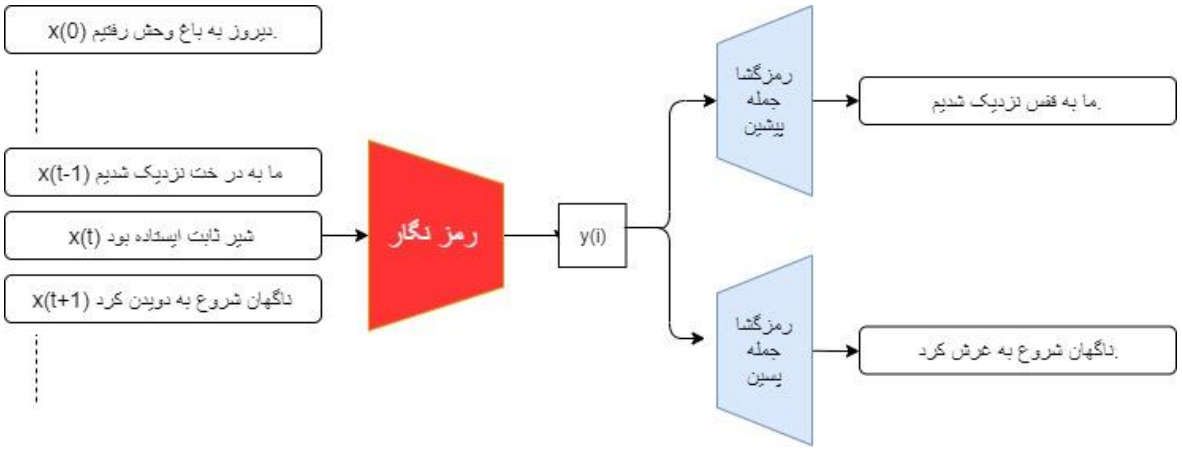
۱-۲- کار های پیشین

نمایش توزیعی کلمه ها یا همان تعبیه کلمات در کاربرد های مختلفی ثابت کرده اند که می توانند ویژگی های کارآمدی ارائه دهند. این روش یکی از اندک کاربرد های موفق یادگیری بدون ناظر است. اولین کار انجام شده در این زمینه در سال ۲۰۰۳ انجام شد (Bengio, 2003). در سال ۲۰۰۸ نیز کولبرت و وستون مقاله ای ارائه کردند که در آن نه تنها این روش به عنوان ابزاری مناسب برای بسیاری از کاربرد ها معرفی شد بلکه یک ساختار شبکه عصبی در آن معرفی شد که پایه این روش و تحقیقات بعدی به حساب می آید (Weston, 2008). گسترش استفاده از این روش و شهرت آن را می توان در سال ۲۰۱۳ و متاثر از کار تحقیقاتی Mikolov دانست که word2vec را معرفی کرد (Le, 2014) . پس از آن نیز مدل های دیگری توسعه داده شدند مثل GloVe و FastText.

در عین حال هنوز بدست آوردن بردار نماینده ای که بتواند به خوبی، یک جمله را با در نظر گرفتن روابط بین کلمات و معنای کلی جمله معرفی کند، یک چالش به حساب می آید. روش هایی مانند SkipToughts یا FastSent از روش های بدون ناظر برای کدگذاری جملات هستند(Kiros, 2015) - (Hill, 2016). البته این روش ها برای آموزش نیاز به حجم داده بسیار بالا دارند و همچنین به زمان و محاسبات بسیار زیادی نیاز دارند. از بین روش های بدون ناظر روش Skip-Thoughts به صورت مختصر توضیح داده می شود.

Skip-Toughts

یک روش بدون ناظر برای بدست آوردن بردار نماینده جملات است که همانند Word2vec پیاده سازی شده است، اما برای جملات توسعه داده شده است. بدین صورت که به جای پیش بینی کلمه های اطراف یک کلمه به پیش بینی جمله های اطراف یک جمله می پردازد. مدل متشکل از دو بخش رمز نگاری و رمز گشایی است که برای پیش بینی جمله قبل و بعد یک جمله آموزش داده می شوند. ساختار آن در شکل قابل مشاهده است.

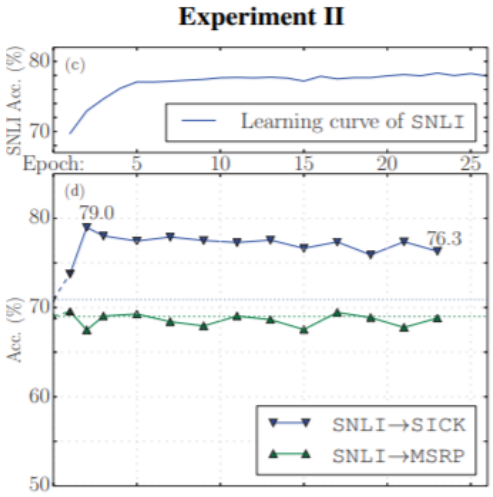


شکل ۱-۱: ساختار و نحوه عملکرد SkipThoughts

(Arora, 2017) نشان می دهد که نتیجه میانگین گیری بردارهای کلمات یک جمله می‌تواند قابل مقایسه با روش های پیچیده ای چون SkipToughts باشد.

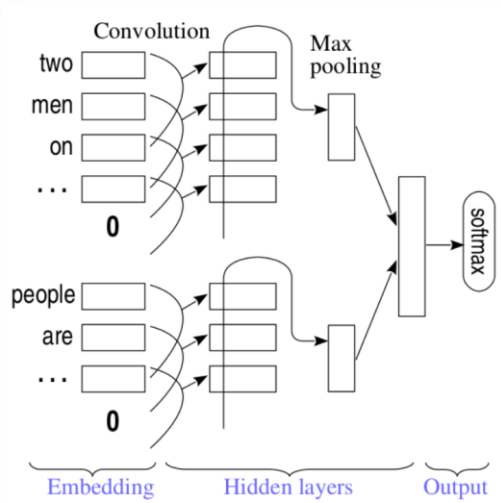
مقاله مورد بررسی (Conneau, 2017)، روشی با ناظر برای بدست آوردن Sentence Embedding ها معرفی می کند. البته قبل از این نیز کار هایی در این زمینه به روش یادگیری با ناظر انجام شده است. مانند (Mou, 2016) . آنها نتیجه گرفتند یادگیری انتقالی در پردازش زبان طبیعی بسیار وابسته به محتوایی است که مدل روی آن آموزش دیده است و همچنین وابسته به محتوایی که مدل بر روی آن ارزیابی می‌شود. در نهایت آنها با طرح چند آزمایش مدل های خود را بر روی مجموعه داده های مختلف آموزش دادند و هر کدام را روی داده های دیگر مورد ارزیابی قرار دادند این داده ها یا برای task احساسات

بودند یا برای task ترجمه. برای مثال نمودار زیر نتیجه آموزش مدل روی داده SLNI و ارزیابی آن روی داده های SICK و MSRP را نشان می دهد. مجموعه داده SICK شامل ۱۰۰۰۰ جفت جمله شامل سه کلاس دقیقا همانند کلاس های موجود در مجموعه داده SLNI است. مجموعه داده MSRP نیز شامل جفت جمله هایی است که برای هر جفت یک لیبل، مشخص کننده اینکه معنای دو جمله یکسان است یا خیر، وجود دارد. همانطور که میبینیم عملکرد مدل انتقالی(آموزش دیده روی داده SLNI) روی داده SICK بهتر است چرا که task مربوط به آن به SLNI شباهت بیشتری دارد.



شکل ۱-۲:ارزیابی مدل آموزش دیده به روش باناظر (Mou, 2016)

بررسی مدل استفاده شده برای رمزنگاری جملات خالی از لطف نیست. مدل استفاده شده به بردار های تعبیه کلمات مربوط به دو جمله عملگر کانولوشن اعمال کرده و سپس از MaxPooling استفاده می کند که خروجی این مرحله Encoding مربوط به هر جمله است در نهایت با ترکیب این دو بردار طبقه بندی روی آن انجام می شود.



شکل ۱-۳: ساختار Encoder استفاده شده (Mou, 2016)

۱-۳- اهداف پروژه

ارائه مکانیزمی به صرفه از نظر حجم محاسبات و زمان مورد نیاز برای آموزش که بتوان با آن بردار نماینده جملات را بدست آورد به طوریکه این بردار ها به خوبی محتوای معنایی و احساسی جمله را در مقایسه با سایر جملات حفظ کنند. چنین مکانیزمی اجازه استفاده از الگوریتم توسعه داده شده در دستگاه های آفلاین را به ما می دهد.

فصل ۲- روش ارائه شده

توضیح روش ارائه شده به دو بخش تقسیم می شود. در بخش اول نحوه آموزش Sentence Encoder ها توسط مجموعه داده SLNI توضیح داده می شود. در بخش دوم ساختار مدل های استفاده شده جهت Encoding توضیح داده می شود در نهایت نتایج عملکرد آنها بررسی می شود.

۲-۱- مجموعه داده استنباطی پردازش زبان طبیعی دانشگاه استنفورد[۱]

این مجموعه داده (Bowman, 2015) شامل ۵۷۰ هزار زوج جمله است. هر زوج جمله شامل یک جمله اصلی و یک جمله است که Hypothesis نامیده می شود. بر اساس ارتباط مفهومی میان دو جمله مذکور ۳ لیبل برای داده ها در نظر گرفته شده است. اگر دو جمله مفهوم یکسانی را برسانند Entailment اگر مفهوم دو جمله در تضاد با یکدیگر باشد Contradiction و اگر جمله Hypothesis شامل مفاهیمی باشد که بر اساس جمله اصلی نمی توان گفت درست است است یا غلط در کلاس Neutral قرار می گیرد. شکل زیر نمونه ای از این مجموعه داده را نشان می دهد.

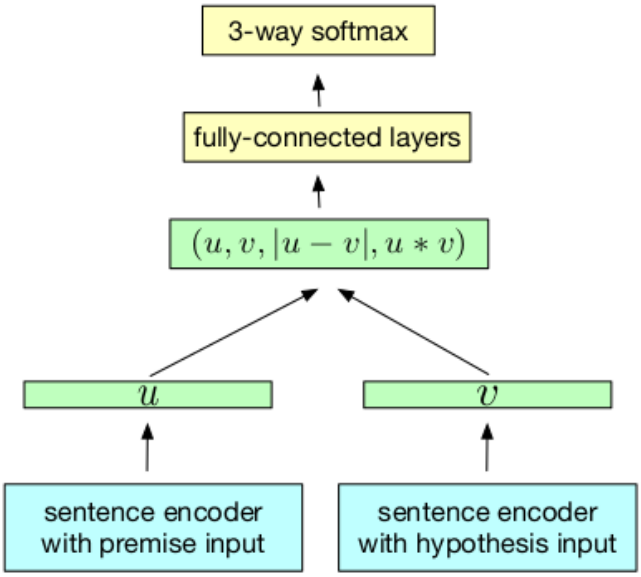
جدول ۲-۱: نمونه ای از مجموعه داده استنفورد(SLNI, 2015)

Text	Judgments	Hypothesis
A man inspects the uniform of a figure in some East Asian country.	contradiction C C C C C	The man is sleeping
An older and younger man smiling.	neutral N N E N N	Two men are smiling and laughing at the cats playing on the floor.
A black race car starts up in front of a crowd of people.	contradiction C C C C C	A man is driving down a lonely road.
A soccer game with multiple males playing.	entailment E E E E E	Some men are playing a sport.
A smiling costumed woman is holding an umbrella.	neutral N N E C N	A happy woman in a fairy costume holds an umbrella.

محققان در مقاله ارائه شده قصد دارند نشان دهند رمز نگار جمله آموزش دیده به روش با ناظر روی داده های با کیفیت NLI می توانند بردار های مناسبی به عنوان نماینده جملات تولید کنند.

[۱] Stanford Natural Language Inference data

ساختار کلی آموزش رمزنگار روی مجموعه داده NLI در شکل ۲-۱ نمایش داده شده است. همانطور که مشاهده می شود جمله اصلی و جمله Hypothesis به مدل رمزنگار ورودی داده می شوند. در مورد ساختار رمزنگار در ادامه توضیح مفصلی ارائه می شود. سپس بردارهای بدست آمده در یک بردار واحد قرار می گیرند به طوریکه بردار بدست آمده بتواند ارتباط بین دو بردار u و v که نماینده جملات هستند را در بر بگیرد. لذا سه المان در این بردار قرار می‌گیرد المان اول بردار حاصل از اتصال دو بردار u و v المان دوم بردار حاصل از تفاضل درایه به درایه دو بردار و المان سوم بردار حاصل از ضرب درایه به درایه دو بردار است. سپس این بردار به یک شبکه عصبی تمام متصل (طبقه بند) ورودی داده می شود و در نهایت احتمالات مربوط به تعلق به هر یک از سه کلاس بدست می آید. لازم به ذکر است برای ورودی دادن کلمات به الگوریتم از مدل از پیش آموزش دیده Glove استفاده شده و تعبیه مربوط به کلمات ایجاد می‌شوند.



شکل ۲-۱: نحوه آموزش مدل ارائه شده به روش با ناظر

۲-۲- ساختار مدل Encoder

در تحقیق انجام شده چندین ساختار شبکه برای تبدیل جملات به بردار های نماینده با طول ثابت معرفی و عملکردشان مورد بررسی قرار گرفته است. در نهایت ساختاری که بهترین عملکرد را داشته معرفی می شود. در ادامه این ساختار ها به صورت مختصر معرفی می شوند.

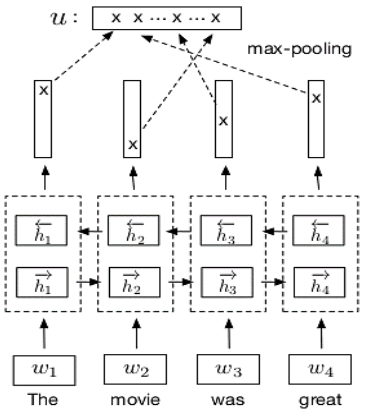
۲-۲-۱- LSTM و GRU

ساده ترین ساختارهای معرفی شده جهت کدگذاری جملات ماژول های LSTM وGRU هستند. برای مثال اگر یک دنباله شامل T کلمه داشته باشیم شبکه T نماینده مخفی بدست می دهد. که آخرین بردار مخفی h_T بردار نماینده جمله است. نگارندگان مقاله از ساختار دیگری به نام BiGRU-last استفاده کردند که در آن آخرین حالت مخفی GRU پیش رو (ورودی آن جمله با ترتیب اصلی است) به همراه GRU وارونه (ترتیب ورودی آن وارونه است) ترکیب می شود و بردار نماینده جمله را می سازد.

$$h_t = \overrightarrow{LSTM}(w_1, \dots., w_T)$$

۲-۲-۲- LSTM دو طرفه با min/max pooling

این ساختار دارای دو LSTM است که یکی از آنها کلمات جمله را با ترتیب اصلی دریافت می کند و دیگری کلمات جمله را با ترتیب وارونه دریافت می‌کند. این دو عنصر به ترتیب LSTM پیش رو و LSTM وارونه نامیده می شوند. حال طبق شکل زیر حالت های مخفی متناظر این دو عنصر در یک بردار کنار هم قرار داده می شوند. روی هر بردار $[\overrightarrow{h_i}, \overleftarrow{h_i}]$، max pooling یا average pooling اعمال می شود. در نهایت خروجی های بدست آمده در کنار هم قرار می گیرند و بردار نماینده جمله یا همان تعبیه جمله را ایجاد می کنند.



شکل ۲-۲: معماری BiDirectional LSTM در (Conneau, 2017)

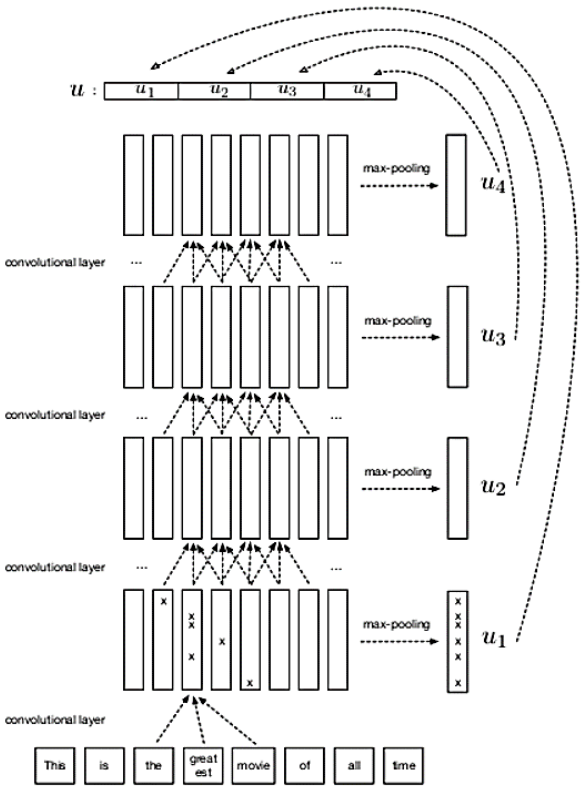
۲-۲-۳- شبکه خود توجه

این شبکه مکانیزم Attention را بر حالت های مخفی BiLSTM اعمال می کند. جزییات مکانیزم توجه به این صورت است که یک تبدیل Affine به هر یک از حالت های مخفی اعمال شده و سپس بردار های محتوا α_i بدست می آید. خروجی نهایی شامل مجموع حالت های مخفی وزن دهی شده با α_i ها است.

$$\begin{aligned} \bar{h}_i &= \tanh(W h_i + b_w) \\ \alpha_i &= \frac{e^{\bar{h}_i^T u_w}}{\sum_i e^{\bar{h}_i^T u_w}} \\ u &= \sum_t \alpha_i h_i \end{aligned}$$

۲-۲-۴- شبکه کانولوشنی سلسله مراتبی

در این تحقیق ساختار کانولوشن سلسله مراتبی نیز مورد ارزیابی قرار گرفت در این ساختار بردار مربوط به کلمات از ۴ لایه کانولشنی عبور کرده به طوریکه پس از هر لایه کانولوشنی Max pooling انجام می شود. سپس خروجی هر چهار مرحله Max pooling با هم ترکیب شده و یک بردار واحد را می سازند که بردار نماینده جمله است.



شکل ۲-۳: معماری شبکه کانولوشنی سلسله مراتبی (Conneau, 2017)

۲–۳- نحوه ارزیابی عملکرد مدل ارائه شده

نگارندگان مقاله کیفیت بردار های جمله تولیدی توسط مدل را در ۱۲ کاربرد مختلف پردازش زبان طبیعی مورد ارزیابی قرار دادند. یکی از مشکلاتی که در هنگام مقایسه عملکرد الگوریتم ها و مدل های مختلف رو برو هستیم یکسان نبودن ابر پارامتر های مربوط به آنهاست برای مثال ممکن است یک encoder برای آموزش طبقه بند از نرخ آموزش ۰,۱ استفاده کند حال آنکه دیگری از نرخ ۰,۰۱ استفاده کند این امر باعث می شود نتایج بدست آمده قابل مقایسه نباشند.

برای حل این مشکل SentEval معرفی شد (Alexis Conneau, 2018) . این ابزار از الگوریتم Adam برای آموزش یک طبقه بند رگرسیون منطقی[۱] استفاده می کند. و می توان از طریق این ابزار، مجموعه داده مربوط به چندین task پردازش زبان را دانلود کرد و سپس عملکرد رمزنگار را بر روی همه آنها ارزیابی کرد. SentEval برای همه این task ها از یک طبقه بند با ابر پارامتر های یکسان استفاده می کند تا نتایج استاندارد و قابل مقایسه باشند.

[۱] Logistic regression

فصل ۳- ارزیابی عملکرد

محققان با مشاهده و بررسی دقت های بدست آمده برای روش ارائه شده و دیگر روش های موجود اعم از باناظر و بدون ناظر ثابت کردند روش با ناظر ارائه شده توسط آنها از دیگر روش ها حتی روش Skip-Thoughts نیز عملکرد بهتری دارد. در واقع آنها توانستند با مجموعه داده بسیار کوچکتر اما لیبل گذاری شده و همچنین صرف زمان و هزینه محاسباتی کمتر به عملکرد بهتری در task های پردازش زبان طبیعی برسند. لذا برای رسیدن به نتایج مطلوب توسط روش با ناظر بایستی task مناسب برای آموزش مدل Encoder انتخاب شود که محققان به این نتیجه رسیدند NLI گزینه مطلوبی است. جدول زیر ثابت کننده این ادعاست.

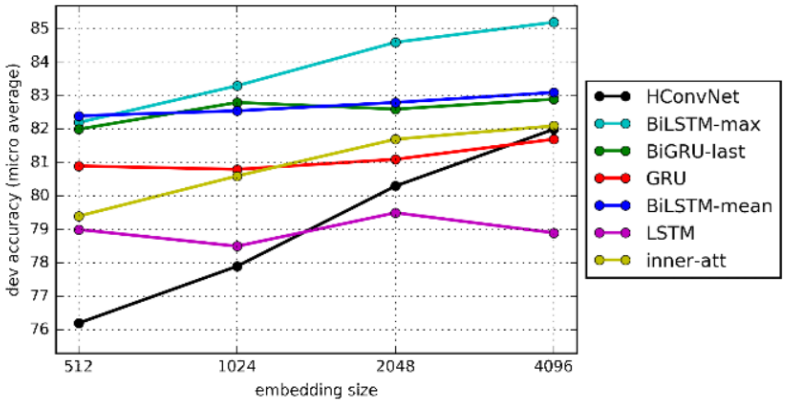
جدول ۳-۱: مقایسه عملکرد الگوریتم های باناظر و بدون ناظر روی مجموعه داده های مختلف NLP

Model	MR	CR	SUBJ	MPQA	SST	TREC	MRPC	SICK-R	SICK-E	STS14
Unsupervised representation training (ordered sentences)										
FastSent	70.8	78.4	88.7	80.6	-	76.8	72.2/80.3	-	-	**.63/.64**
FastSent+AE	71.8	76.7	88.8	81.5	-	80.4	71.2/79.1	-	-	.62/.62
SkipThought	76.5	80.1	93.6	87.1	82.0	**92.2**	**73.0/82.0**	**0.858**	82.3	.29/.35
SkipThought-LN	**79.4**	**83.1**	**93.7**	**89.3**	82.9	88.4	-	**0.858**	79.5	.44/.45
Supervised representation training										
CaptionRep (bow)	61.9	69.3	77.4	70.8	-	72.2	73.6/81.9	-	-	.46/.42
DictRep (bow)	76.7	78.7	90.7	87.2	-	81.0	68.4/76.8	-	-	**.67/.70**
NMT En-to-Fr	64.7	70.1	84.9	81.5	-	82.8	69.1/77.1	-		.43/.42
Paragram-phrase	-	-	-	-	79.7	-	-	0.849	83.1	**.71**/ -
BiLSTM-Max (on SST)†	(*)	83.7	90.2	89.5	(*)	86.0	72.7/80.9	0.863	83.1	.55/.54
BiLSTM-Max (on SNLI)†	79.9	84.6	92.1	**89.8**	83.3	**88.7**	75.1/82.3	**0.885**	**86.3**	.68/.65
BiLSTM-Max (on AllNLI)†	**81.1**	**86.3**	**92.4**	**90.2**	**84.6**	88.2	**76.2/83.1**	**0.884**	**86.3**	**.70/.67**

در مرحله بعد، طبیعتا ساختار مدل رمزنگار نیز اثر به سزایی در تولید تعبیه جملات با کیفیت دارد. نمودار شکل ۳-۱ اثبات کننده این ادعاست. ساختار BiLSTM به بهترین عملکرد دست یافت همچنین نتیجه گرفته شد هرچه ابعاد فضای تعبیه[۱] یا همان طول بردار نماینده تولید شده برای هر جمله بیشتر باشد دقت بالاتر است. این همان فرضیه معروف (Cover, 1965) است که می گوید یک مسئله تفکیک پذیر غیر خطی در ابعاد بالاتر احتمال بیشتری دارد که به صورت خطی قابل جداسازی باشد. نمودار زیر میزان دقت بدست آمده روی مجموعه داده SNLI را بر حسب طول بردار رمزنگاری شده[۲] نشان می دهد. این نمودار به تفکیک ساختار های مختلف مدل رمزنگار رسم شده است.

[۱] Embedding

[۲] Embedding vector length



شکل ۳-۱: دقت بدست آمده روی داده SLNI بر حسب سایز بردار embedding و به تفکیک ساختار های مختلف Encoder - (Conneau, 2017)

با توجه به جدول ۳-۱ عملکرد روش با ناظر ارائه شده از بهترین مدل آموزش دیده به روش بدون ناظر، که SkipThoughts-LN نام دارد، بهتر است. لازم به ذکر است محققان علاوه بر آموزش مدل با داده SLNI در آزمایشی دیگر مدل را با مجموعه داده AllNLI نیز آموزش دادند. این مجموعه داده اجتماع دو مجموعه داده SLNI و MultiNLI است که شامل ۱ میلیون داده لیبل گذاری شده است. عملکرد مدل آموزش دیده روی ALLNLI از همه مدل های دیگر بهتر است.

فصل ۴- جمع بندی و نظرات (مربوط به گزارش اولیه)

مقاله ارائه شده توانایی روش های با ناظر برای استخراج بردار نماینده جملات را آشکار کرد. در مقالات گذشته به تاثیر task انتخاب شده برای آموزش مدل بر عملکرد آن اشاره شده بود. در این تحقیق می توان گفت task مناسب برای آموزش با ناظر مشخص شد. علاوه بر آن محققان ساختار مدل رمزنگار مناسب را نیز با انجام چند آزمایش مشخص کردند.

از مزیت های این روش نسبت به روش های بدون ناظر مثل SkipThoughts می توان به این موارد اشاره کرد:

- نیازمند حجم داده کمتر برای آموزش
- زمان آموزش بسیار کمتر(۱ روز) در مقایسه با زمان آموزش بالا (چندین روز)
- هزینه محاسباتی کمتر (چه در زمان آموزش چه در زمان Inference)
- می توان به اجرای آن به صورت آفلاین و بلادرنگ فکر کرد در حالیکه روش های بدون ناظر بایستی حتما روی سیستم قوی بر پایه Cloud اجرا شوند.

از معایب این روش نسبت به روش های بدون ناظر می توان به موارد زیر اشاره کرد:

- هزینه بر بودن جمع آوری داده با کیفیت
- چون به صورت باناظر آموزش دیده می شود ممکن است بایاس موجود در مجموعه داده ها را یاد بگیرد و در تعمیم[۱] آن تاثیر منفی بگذارد.
- نیاز با استفاده از تعبیه کلمات از پیش آموزش دیده مثل FastText

برای بهبود روش ارائه شده می توان ساختار مدل رمزنگار آن را بهبود بخشید تا علی رغم داشتن پارامتر های کمتر دقت بهتری داشته باشد. این موضوع قابلیت اجرای آفلاین آن بر روی دستگاه های همراه را بهبود می بخشد. همچنین می توان برای افزایش تعمیم مدل آن را به صورت چند وظیفه[۲] ای آموزش داد.

[۱] Generalization

[۲] Multi-task

فصل ۵- بهبود الگوریتم

جهت بهبود الگوریتم گزینه های مختلفی وجود دارد. از جمله تغییر مدل استفاده شده، تغییر مدل تعبیه کلمات، افزایش داده های آموزشی و استفاده از ایده های ابتکاری در پیش پردازش داده ها.
با توجه به امکانات و زمان موجود، تغییرات انجام شده بر الگوریتم ارائه شده در مقاله شامل مراحل زیر است:

- تغییر در ساختار مدل و استفاده از مدل ESIM
- استفاده از تعبیه کلمات FastText بجای GloVe

دلیل استفاده از FastText بررسی این موضوع بود که آیا تفاوت چندانی بین استفاده از این مدل و مدل GloVe وجود دارد یا نه. در فصل پایانی نتایج حاصل از استفاده این دو مدل جهت تولید تعبیه کلمات گزارش می شود.

۵-۱- بهبود یافته LSTM برای کاربرد استنباط در زبان طبیعی (ESIM)

مدلی که برای بهبود عملکرد الگوریتم ارائه شده معرفی می شود، بر ا ساس کار تحقیقاتی (Chen, 2017) ا ست. این مدل به طور خاص برای بهبود اطلاعات ا ستنباطی مدل از متن طراحی شده ا ست و محققان توانسته اند به دقت ۸۸٪ بر داده های ارزیابی مجموعه داده SNLI برسند. این دستاورد، بهبود ۳ در صدی را در مقایسه با مدل ا صلی نشان می دهد و می تواند نوید بخش بهبود کیفیت تعبیه جملات تولید شده توسط مدل باشد و توانایی آن را در Task های دیگر پردازش زبان بهبود بخشد. مدل ESIM از سه بخش ا صلی تشکیل شده ا ست : ۱. رمزنگاری ورودی ۲. مدل سازی ا ستنباط محلی ۳. بد ست آوردن استنباط نهایی. این سه مورد در ادامه به شکل کامل توضیح داده خواهند شد.

۵–۱–۱- رمز نگاری ورودی

رمزنگاری ورودی به کمک لایه BiLSTM بر روی دو جمله اصلی[۱] و جمله فرضیه[۲] اعمال می شود. لذا اگر جمله اصلی را a طول آن را l_a ، جمله فرضیه را b و طول آن را l_b بنامیم، داریم:

$$\bar{a}_i = BiLSTM(a, i), \forall i \in [1, \dots, l_a]$$
$$\bar{b}_j = BiLSTM(b, j), \forall j \in [1, \dots, l_b]$$

در واقع در این بخش وظیفه BiLSTM یادگیری نمایش هر کلمه در جمله مربوط به آن است. همانطور که در فصل ۳ توضیح داده شد، این کار بر اساس محتوا و کلمات اطراف کلمه اصلی صورت می پذیرد.

۵–۱–۲- مدل سازی استنباط محلی

برای درک بهتر ارتباط استنباطی میان دو جمله اصلی و فرضیه نیاز به داشتن اطلاعاتی در مورد ارتباط معنایی میان کلمات بکار رفته در دو جمله داریم. این ارتباط چون محدود به جفت کلمات موجود در دو جمله می شود یک ارتباط محلی است. برای مدل کردن عملیات استنباط محلی بایستی از مکانیزم توجه به صورت نرم یا سخت[۳] استفاده شود. در مقاله مذکور (Chen, 2017) ، از روش نرم استفاده شده است به این صورت که برای بدست آوردن وابستگی معنایی بین دو کلمه، حالت های مخفی مربوط به آنها (ناشی از ورودی دادن کلمات جمله به BiLSTM) در هم ضرب داخلی می شوند.

$$e_{ij} = \bar{a}_i^T \bar{b}_j$$

حال چون ورودی الگوریتم دو جمله هستند و پس از عبور از BiLSTM به دو بردار $\bar{a}_i$ و $\bar{b}_j$ تبدیل می شوند، برای بدست آوردن بردار مربوط به استنباط محلی از رابطه زیر استفاده می کنیم:

$$\tilde{a}_i = \sum_{j=1}^{l_b} \frac{\exp(e_{ij})}{\sum_{k=1}^{l_b} \exp(e_{ik})} \bar{b}_j, \forall i \in [1, \dots, l_a]$$
$$\tilde{b}_j = \sum_{i=1}^{l_a} \frac{\exp(e_{ij})}{\sum_{k=1}^{l_a} \exp(e_{kj})} \bar{a}_i, \forall j \in [1, \dots, l_b]$$

---

[۱] Premise

[۲] Hypothesis

[۳] Soft or Hard Attention

این به این معناست که دو بردار حاصل از استنباط محلی خواهیم داشت. برای مثال اگر بردار $\tilde{a}$ را در نظر بگیریم ، به ازای هر $\tilde{a}_i$ ، هر درایه ای از بردار $\bar{b}_j$ که با $\bar{a}_i$ مرتبط باشد در نظر گرفته شده و در خروجی اثر می گذارد.

جهت بهبود اطلاعات و کیفیت بردار حاوی ا ستنباط محلی، بردار حا صل از تفریق و ضرب درایه به درایه دو بردار $<\bar{a}_i, \tilde{a}_i>$ نیز بدست آمده و در کنار آنها قرار می گیرد. به بیان واضح تر داریم:

$$m_a = [\bar{a}_i, \quad \tilde{a}_i, \quad \bar{a}_i - \tilde{a}_i, \quad \bar{a}_i \odot \tilde{a}_i]$$

بطور مشابه برای بردار مربوط به جمله فرضیه داریم:

$$m_b = [\bar{b}_j, \quad \tilde{b}_j, \quad \bar{b}_j - \tilde{b}_j, \quad \bar{b}_j \odot \tilde{b}_j]$$

۵-۱-۳- استنباط نهایی

در این مرحله m_a, m_b به biLSTM ورودی داده می شوند تا استنباط کلی مدل از هر جمله بدست آید. در مرحله آخر max_pooling به همراه avg_pooling به دو بردار اعمال شده و نتایج در یک ماتریس به نام v کنار هم قرار می گیرند. به بیان ریاضی داریم:

$$\mathbf{v}_{a,\text{ave}} = \sum_{i=1}^{\ell_a} \frac{\mathbf{v}_{a,i}}{\ell_a}, \quad \mathbf{v}_{a,\max} = \max_{i=1}^{\ell_a} \mathbf{v}_{a,i},$$

$$\mathbf{v}_{b,\text{ave}} = \sum_{j=1}^{\ell_b} \frac{\mathbf{v}_{b,j}}{\ell_b}, \quad \mathbf{v}_{b,\max} = \max_{j=1}^{\ell_b} \mathbf{v}_{b,j},$$

$$\mathbf{v} = [\mathbf{v}_{a,\text{ave}}; \mathbf{v}_{a,\max}; \mathbf{v}_{b,\text{ave}}; \mathbf{v}_{b,\max}].$$

بردار $\mathbf{v}_a$ یا در واقع همان بردار تعبیه جمله a است که دارای ۱۰۲۴ بعد است و بایستی یادگرفته شود. برای آموزش این Encoder یک طبقه بند متصل به بردار v با تابع هزینه Categorical Cross entropy تعریف شده و مدل به صورت Supervised آموزش داده می شود.

فصل ۶- جزئیات پیاده سازی الگوریتم و آموزش آن

برای آموزش مدل از ســـرویس رایگان گوگل کولب اســـتفاده شـــد. مراحل پیاده ســـازی و آموزش الگوریتم به شرح زیر است:

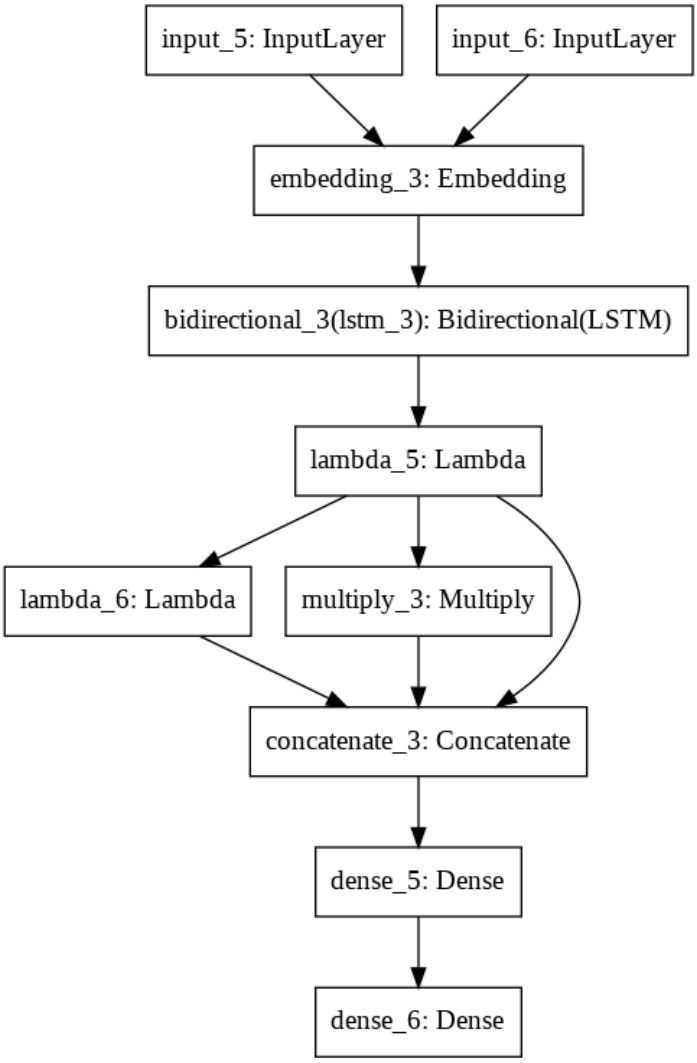
۱. بارگذاری مجموعه داده SNLI
۲. پیش پردازش مجموعه داده
   a. حذف داده های ناقص
   b. حذف داده های بدون لیبل
   c. رمزگذاری One-hot لیبل ها
   d. Tokenize جملات
   e. دانلود و خواندن مدل تعبیه کلمات (FastText و GloVe)
   f. بدست آوردن Embedding matrix
۳. پیاده سازی مدل ها
   a. مدل اصلی ارائه شده در مقاله
   b. مدل بهبود یافته (ESIM)
۴. آموزش مدل ها
۵. گزارش، رسم و تحلیل نتایج

از این بخش ها تنها بخش ۳ ، ۴ و ۵ نیاز به توضیح دارند. قیه بخش ها در داخل کد مشخص و کامنت نویسی شده است. بخش ۵ در فصل آینده بررسی و توضیح داده می شود.

۱-۶- پیاده سازی مدل ها و آموزش

دو مدل پیاده سازی و آموزش داده شدند. مدل اصلی معرفی شده در مقاله و مدل ESIM. لازم به ذکر است مدل ESIM یکبار با استفاده از GloVe و یکبار با استفاده از FastText آموزش داده شد تا نتایج مقایسه شوند. تمامی مدل ها توسط کتابخانه Keras پیاده سازی شده اند.

ساختار مدل اصلی



شکل ۶-۱: ساختار مدل اصلی رسم شده با Keras

همانطور که در فصل سوم به طور مفصل راجع به این ساختار صحبت شد، این مدل دو جمله اصلی و فرضیه را دریافت کرده و به BiLSTM ورودی می دهد که اگر خروجی آنرا برای دو جمله u و v بنامیم. بردار نهایی به شکل زیر حاصل می شود که شامل ۴۰۹۶ بعد است:

در ادامه این بردار یک طبقه بند با یک لایه مخفی ۱۲۸ نورون و یک خروجی ۳ کلاسه قرار می گیرد. این مدل با batch_size = 256 و به کمک الگوریتم بهینه سازی Adam با هایپرپارامتر های پیش فرض آموزش داده شده است. تابع هزینه Categorical Cross entropy است. در پایان آموزش شبکه تا لایه lstm_3 به عنوان تعبیه کننده جملات بکار می رود.

ساختار مدل بهبود یافته – ESIM

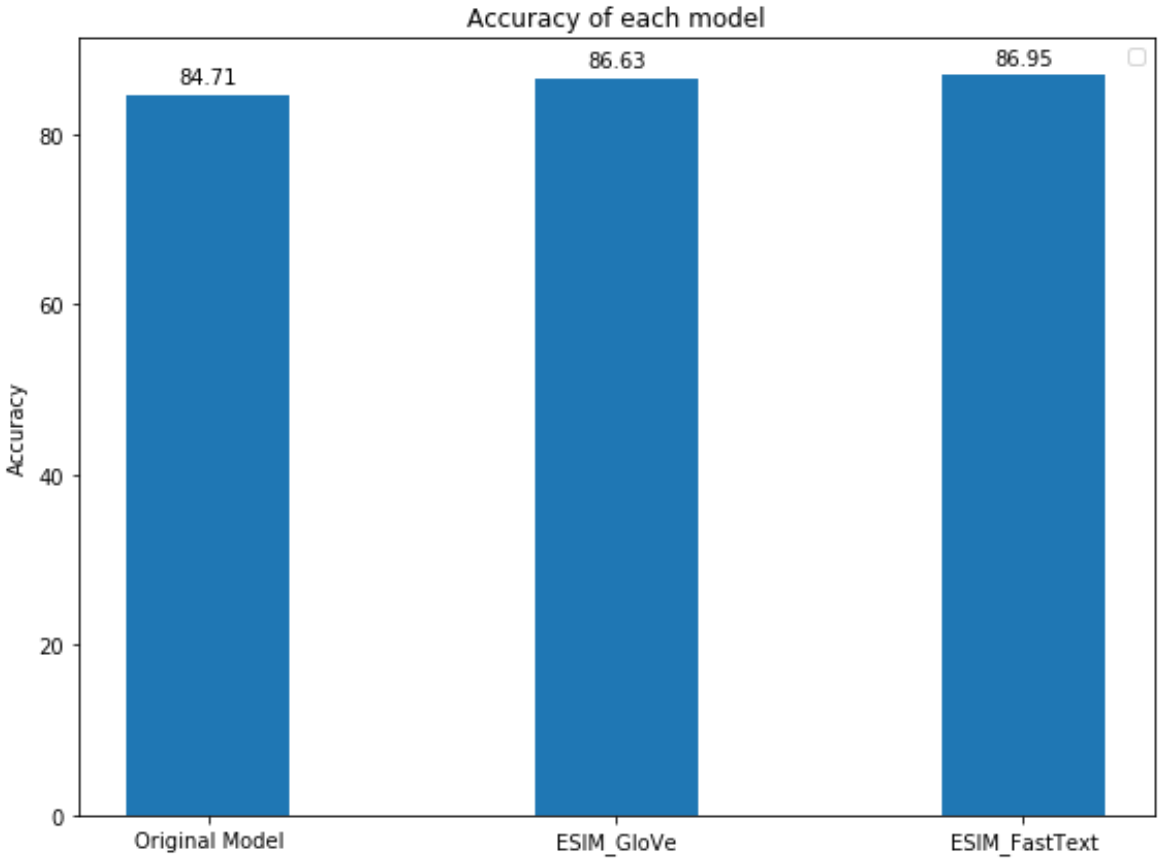
شکل ۶-۲: ساختار مدل بهبود یافته ESIM رسم شده توسط Keras

این مدل نیز مطابق توضیحات بخش ۵-۱ پیاده سازی شده است. لازم به ذکر است برای پیاده سازی این مدل نیاز به تعریف یک لایه جدید به نام DotProductAttention بود تا عملیات Local Inference توضیح داده شده در بخش ۵-۱-۲ انجام گیرد.

بردار نهایی بدست آمده از مدل Encoder که همان خروجی لایه Concatenate در شکل ۶-۲ است، به یک طبقه بند ورودی داده می شود که یک لایه مخفی با ۲۵۶ نورون و لایه خروجی با ۳ نورون است. الگوریتم بهینه سازی Adam با ابرپارامتر های پیش فرض است و تابع هزینه Categorical Cross Entropy در نظر گرفته شده است.

فصل ۷- نتایج و تحلیل آنها

دقت بدست آمده توسط ۳ مدل اصلی،ESIM-GloVe و ESIM-FastText روی مجموعه داده NLI ، در نمودار زیر گزارش شده است:



شکل ۷-۱: نمودار دقت مدل های آموزش داده شده

بالاترین دقت مربوط به مدل ESIM با تعبیه کلمات FastText است. **این مدل دقت مدل اصلی را به میزان ۲,۲۴ درصد افزایش داده است که نشان میدهد استفاده از ساختار جدید شامل Local Inference و Enhanced Local Inference اثر بخش بوده است.** همچنین استفاده از مدل GloVe یا FastText تفاوت آنچنانی در نتایج پایانی ندارد. درحدول زیر تعداد پارامتر ها و حجم مدل ها نیز با یکدیگر مقایسه می شوند.

جدول ۷-۱: تعداد پارامتر ها و حجم مدل ها

نام مدل	Original(BiLSTM-Max)	ESIM
تعداد پارامتر ها	۳ ۸۵۴ ۸۵۱	۱۰ ۶۷۵ ۷۱۵
حجم مدل	۸۴ مگابایت	۱۶۲ مگابایت

۱–۷- عملکرد مدل آموزش دیده روی مجموعه داده دیگر (یادگیری انتقالی)

SICK-E یکی از مجموعه داده های موجود در NLP است که همانند SNLI برای Task استنباط تهیه شده است. این مجموعه داده شامل ۱۰ هزار جفت جمله اصلی و فرضیه است، به همراه لیبل سه کلاسه Neutral، Entailment و Contradiction . **دقت بدست آمده با مدل بهبود یافته نسبت به مدل اصلی و آنچه در مقاله گزارش داده شده است ۲,۴۳ درصد بهبود داشته است.**

برای انتقال یادگیری تمامی وزن های مدل ثابت شدند و فقط وزن های بخش طبقه بند در زمان آموزش اجازه تغییر داشتند. طبقه بند نیز دارای ۲۵۶ نورون در لایه مخفی و سه نورون در لایه خروجی است. تابع هزینه Categorical Cross entropy و بهینه ساز Adam انتخاب شده است.

جدول ۷-۲: عملکرد مدل ها روی داده SICK

نام مدل	Original(BiLSTM-Max)	ESIM
دقت داده تست SICK	%۸۶,۳	%۸۸,۷۳

- در نهایت از مدل آموزش دیده شده، مدلی ایجاد شد تا با دریافت یک جمله بردار مربوط به آن را برگرداند.

References

(n.d.). Retrieved from https://en.wikipedia.org/wiki/De_Bothezat_helicopter

Alexis Conneau, D. K. (2018). SentEval: An Evaluation Toolkit for Universal Sentence Representations. *arXiv preprint arXiv:1803.05449*.

Arora, S. L. (2017). A simple but tough-to-beat baseline for sentence embeddings. *International Conference on Learning Representations.*

Bengio, Y. D. (2003). A neural probabilistic language model. *Journal of machine learning research*, 1137-1155.

Bowman, S. R. (2015). A large annotated corpus for learning natural language. arXiv preprint arXiv:1508.05326.

Chen, Q. Z. (2017). Enhanced LSTM for Natural Language Inference. . *In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, (pp. (Volume 1: Long Papers) (pp. 1657-1668)).

Conneau, A. K. (2017). Supervised learning of universal sentence representations from natural language inference data. arXiv preprint arXiv:1705.02364.

Cover, T. M. (1965). Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE transactions on electronic computers*, (3), 326-334.

Hill, F. C. (2016). Learning distributed representations of sentences from unlabelled data. arXiv preprint arXiv:1602.03483.

Kiros, R. Z. (2015). Skip-thought vectors. *In Advances in neural information processing systems*, 3294-3302.

Le, Q. &. (2014). Distributed representations of sentences and documents. *In International conference on machine learning*, (pp. 1188-1196).

M. Marelli, S. M. (2014). A SICK cure for the evaluation of compositional distributional semantic models. *Proceedings of LREC 2014*, (pp. ELRA, 216-223.). Reykjavik (Iceland).

Mou, L. M. (2016). How transferable are neural networks in nlp applications? arXiv preprint arXiv:1603.06111.

SLNI. (2015). Retrieved from Stanford NLP: https://nlp.stanford.edu/projects/snli/

Weston, R. C. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. *In Proceedings of the 25th international conference on Machine learning* (pp. 160–167). ACM.

Iran University of Science and Technology
Faculty of Computer Engineering
Department of Artificial Intelligence

Supervised Learning of Universal Language Representations from Natural Language Inference Data

First Report

By:
Taha Samavati

Supervisor:
Dr. Behrooz Minaee

Fall 2019