مشخصات دانشجو:

نام و نام خانوادگی:	مهرداد رفیعی پور			شماره دانشجویی:	9821580207
مقطع:	کارشناسی ارشد	رشته:	مهندسی کامپیوتر	گرایش:	نرم افزار

مشخصات استادان راهنما و مشاور:

سمت	نام و نام خانوادگی	درصد همکاری	دانشگاه/مؤسسه	مرتبه علمی	تخصص مرتبط با پایان نامه
استاد راهنمای اول	دکتر جواد سلیمی سرتختی		دانشگاه کاشان	استادیار	
استاد مشاور اول	دکتر فرشته دهقانی		دانشگاه کاشان	استادیار	

عنوان پیشنهادی پژوهش:

عنوان فارسی:	**ارائه یک مدل برای تبدیل خودکار متن زبان طبیعی به پرس‌وجوی SQL با استفاده از شبکه‌های عصبی عمیق**
Title:	**Proposing a model for auto-conversion of natural language text to SQL query using deep neural networks**

کلمات کلیدی (3 تا 5 مورد) **:**

کلمات کلیدی فارسی:	**شبکه عصبی عمیق، پردازش زبان طبیعی، پایگاه‌داده، متن به SQL**
Keywords:	Deep Neural Network, Natural language Processing, Database, Text to Sql

1-شرح مختصری از موضوع کلی و زمینهٔ اصلی پژوهش:

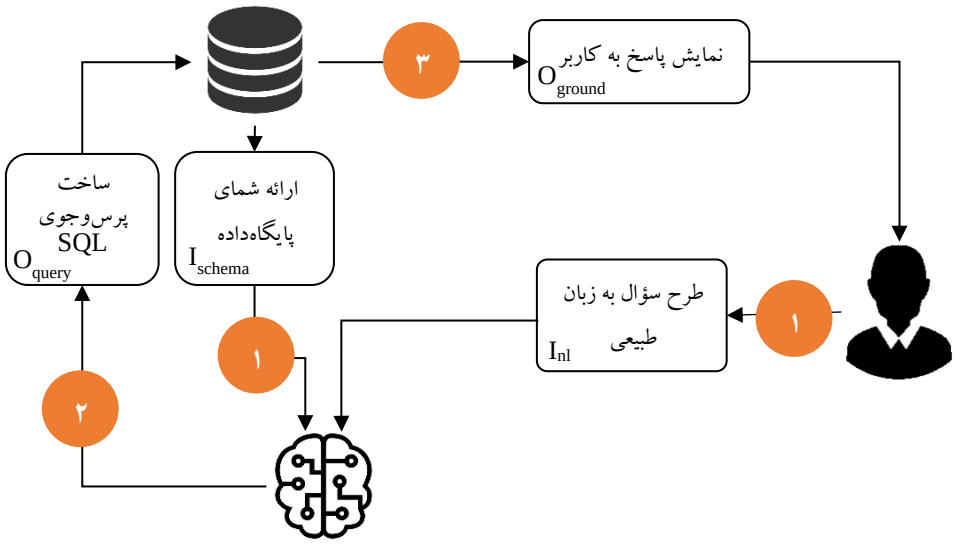
امروزه حجم زیادی از داده‌های دیجیتال در پایگاه‌داده‌ها نگهداری می‌شوند. دسترسی به این داده‌ها نیازمند کسب اطلاعات درمورد ساختار پایگاه‌داده‌ها و یادگیری روش کارکردن با آن‌هاست؛ به همین دلیل دسترسی به این داده‌ها توسط افراد غیرمتخصص دشوار است. هدف ما در امر تبدیل متن به پرس‌وجو این است که دسترسی افراد غیرمتخصص را به این حجم وسیع از داده‌ها فراهم کنیم.

ازطرفی پاسخ بخشی از سؤالاتی که از موتورهای جستجو پرسیده می‌شوند، بصورت فکتوئید[1] است. یعنی پاسخ آن بصورت چند کلمه و کوتاه است و هدف پرسشگر، پیدا کردن لینک، پاراگراف یا موارد دیگر نیست. به همین دلیل می‌توان بجای جستجوی پاسخ در متون طولانی، خلاصه‌ای از اطلاعات را درمورد موضوعات مختلف، در یک پایگاه‌داده ذخیره کرد و با تبدیل متن‌به‌پرس‌وجو، یک کوئری برای دریافت پاسخ ایجاد کرد. با تبدیل مسئله از جستجو در پاراگراف‌ها به جستجو در پایگاه‌داده، می‌توان هزینه‌ی محاسبات را برای موتور جستجو کاهش داد که این خود بیانگر اهمیت وظیفه‌ی متن‌به‌پرس‌وجو است [1]. همچنین در پایگاه‌داده‌های تجاری، می‌توان از سیستم‌های تبدیل متن به پرس‌وجو استفاده کرد تا مدیران سیستم بتوانند بدون نیاز به یادگیری مفاهیم پایگاه‌های داده، اطلاعات کسب‌وکار خود را در محیطی کاربرپسند مشاهده کنند [2].

در مسئله‌ی تبدیل متن به پرس‌وجو، مطابق شکل 1 کاربر سؤالی را به زبان طبیعی مطرح می‌کند که پاسخ آن درون یک پایگاه‌داده است. ما می‌خواهیم ضمن تبدیل کردن سؤال کاربر به پرس‌وجو، پاسخ را از پایگاه‌داده دریافت کرده و آن را بصورت فکتوید به کاربر نمایش دهیم. این روند در شکل 1 نشان داده شده است.

[1] Factoid

درواقع وظیفه‌ی ما طراحی یک مدل شبکه‌ی عصبی است که سؤال کاربر I_{nl} و شمای پایگاه‌داده I_{schema} را دریافت کرده و در پاسخ یک پرس و جو O_{query} بر روی پایگاه‌داده‌ی مذکور ایجاد کند و پاسخ دریافتی از پایگاه‌داده O_{ground} را به کاربر نمایش دهد.



شکل 1 فرایند عملکرد مسئله‌ی متن به SQL

به منظور انجام وظیفه‌ی مذکور، لازم است که مدل شبکه‌ی عصبی را بر روی دیتاست مشخصی آموزش دهیم. هدف اصلی ما طراحی مدلی است که بتواند حل مسئله‌ی مذکور را یاد بگیرد؛ یعنی در مواجه با سؤال دیده نشده و همچنین ساختار جدید پایگاه‌داده، بتواند پرس‌وجویی ایجاد کند که پاسخ صحیح را به کاربر نمایش دهد. به این منظور مدل باید رابطه‌ی میان کلمات درون پرسش کاربر با ستون‌های پایگاه‌داده را کشف کند و همچنین درک مناسبی از روابط میان جداول پایگاه‌داده داشته باشد. این مسئله می‌تواند با آموزش مجدد مدل بر روی دیتاست هدف و یا بدون آموزش جدید صورت گیرد. گرچه ثابت شده آموزش مجدد مدل حداقل به عملکرد بهتر مدل کمک می‌کند [3].

2- پیشینه‌ی انجام کارهای قبلی مرتبط با موضوع پژوهش:

روش‌های گوناگونی برای حل مسئله وجود دارد. این روش‌ها بر اساس شیوه‌ی تولید خروجی، روش آموزش و اساس معماری شبکه دسته بندی شده‌اند. در این بخش ابتدا هر دسته را تعریف می‌کنیم و سپس به معرفی روش‌های برتر برای حل مسئله‌ی متن به پرس‌وجو می پردازیم.

۲-۱ تولید خروجی

یک دسته بندی مربوط به چگونگی تولید ترتیب خروجی[1] است. عمدتا ترتیب خروجی به سه روش تولید می‌شود:

مبتنی بر تولید[2]: در روش مبتنی بر تولید، پرس‌وجوی SQL را بر مبنای مدل Sequence-to-Sequence تولید می‌کنند. نقطه ضعف این مدل، مکانیزم توجه[3] و کپی کردن توکن‌هایی از ورودی - که معمولا توکن‌های موجودیت‌ها هستند - به سمت خروجی است. برای مثال چون فضای خروجی softmax (لایه‌ی آخر که پرس‌وجو را تولید می‌کند) برای امر تبدیل متن به پرس‌وجو بی‌دلیل گسترده‌است، Seq2SQL این فضا را محدود به توکن‌های سؤال، ستون‌های جدول و توکن‌های دستورالعمل SQL می‌کند [4]. این مدل‌ها عمدتا مشکل جابجایی کلمات درون پرس‌وجوی SQL را دارند زیرا چنین مدلی نمی‌تواند ترتیب دستورالعمل[4] SQL را به خوبی پیاده‌سازی کند.

مبتنی بر طرح[5]: به دلیل ذکر شده در بالا، در برخی از کارها از طرح‌های از پیش تعیین شده به منظور محدود سازی فضای خروجی شبکه و افزایش دقت مدل استفاده می‌کنند. اینگونه مدل‌ها مبتنی بر طرح نام دارند. مراجع [5 - 6] از خروجی مبتنی بر طرح استفاده کرده‌اند. در مدل‌های مذکور، شبکه ابتدا یک طرح از میان طرح‌های موجود را انتخاب کرده و سپس با توجه به طرح اقدام به تولید توکن می‌کند.

مبتنی بر پرکردن اسلات[6]: برای حل مشکل مطرح شده در بخش مبتنی بر تولید، گاهی روش مبتنی بر اسلات استفاده می‌شود. در این روش یک قالب مشخص برای پرس‌وجو درنظر گرفته می‌شود. این مدل، پرس‌وجوی SQL را به تعدادی زیرمدل تقسیم کرده و پس از تقسیم بندی، چند شبکه آموزش داده می‌شود تا هرکدام جاهای خالی مربوط به بخش خود در طرح ایجاد شده را پر کند. مرجع [7] از این روش استفاده کرده است.

۲-۲ روش آموزش

مدل‌های مطرح شده برای حل مسئله به دو شیوه آموزش داده می‌شوند؛ مدل‌های مبتنی بر یادگیری تقویتی و مدل‌های مبتنی بر یادگیری با ناظر.

مبتنی بر یادگیری تقویتی[7]

چون داده‌های برچسب‌دار، منبعی کم‌یاب و هزینه‌بر است، مطلوب است در مدل‌ها تا جای ممکن از روش‌هایی استفاده شود که نیاز به این منابع را به حداقل برساند. درصورت استفاده از روش‌های مبتنی بر یادگیری تقویتی، می‌توان از بازه‌ی گسترده‌تری از داده‌ها استفاده کرد. Seq2SQL یکی از مدل‌هایی است که از یادگیری تقویتی در فرایند آموزش خود بهره می‌برد [4]. یکی از چالش‌های یادگیری تقویتی گیرکردن مدل در کمینه محلی[8] است که مدل Seq2SQL نیز گرفتار آن بود. به این منظور SeqPolicyNet معرفی شد که از مفهوم پاداش بونی[9] استفاده می‌کرد [7]. پاداش بونی جایزه‌ای برای انتخاب حالت‌هایی که دیده نشده‌اند انتخاب می‌کرد تا مدل به انتخاب حالت‌های جدید – و همچنین فرار از مینیموم محلی – ترغیب شود.

[1] Output Sequence
[2] Generation-Based
[3] Attention mechanism
[4] Syntax
[5] Sketch-Based
[6] Slot Filling
[7] Reinforcement Learning
[8] Local Minima
[9] Reward Boni

مبتنی بر یادگیری با ناظر[1]

عمده‌ی روش‌هایی که بالاترین دقت را در وظیفه‌ی تبدیل متن به SQL دارند مبتنی بر یادگیری با ناظر هستند. درصورتی که دیتاست حاوی مقادیر هدف یا برچسب باشد، بهتر است از روش با ناظر استفاده شود. عمده ضعف روش‌های باناظر در وظیفه‌ی تبدیل متن به پرس‌وجو گستردگی فضای خروجی است. راهکار حل این مشکل استفاده از دیکشنری خروج محدود است [4]. در این دیکشنری ازکلیدواژه‌های SQL، نام ستون‌ها و کلمات موجود در متن سؤال و همچنین اعداد ثابت موجود در مثال‌های آموزشی استفاده می‌گردد. با استفاده از این روش، چون مدل تنها کلمات موجود درون دیکشنری را انتخاب می‌کند، دقت میزان چشم‌گیری افزایش می‌یابد. اما اگر بخواهیم دیتاست جدیدی استفاده کنیم، مدل باید مجددا آموزش ببیند.

۲-۳ اساس معماری شبکه

ساختار درونی مدل‌های مطرح شده بر پایه‌ی دو نوع معماری بنا شده‌اند.

مدل‌های مبتنی بر LSTM[2]

این نوع شبکه از مدلی بهبود داده‌شده از شبکه‌ی عصبی بازگشتی است که برای پردازش داده‌های ترتیبی مانند صدا، تصویر و متن استفاده می‌شود. در این شبکه خروجی در هر مرحله زمانی وابسته به خروجی و حالت مخفی در مرحله زمانی قبلی است. مدل‌های ارائه شده در مقالات [4] و [7 - 11] از این معماری بهره بردند. در معماری IRNet از LSTM در بخش رمزگشا برای تولید توکن‌ها و در بخش رمزنگار[3] از معماری ترانسفورمر استفاده شد [11].

مدل‌های مبتنی بر ترانسفورمر[4]

شبکه‌های LSTM دو ایراد عمده داشتند؛ گرادیان محو شونده[5] و گرادیان انفجاری[6]. همچنین درک محتوای جمله بصورت حقیقی دوطرفه نبود، بلکه از تجمیع ترتیب راست به چپ و چپ به راست بدست می‌آمد. به همین علت شبکه‌های ترانسفورمر [12] معرفی شدند که تنها مبتنی بر مکانیزم توجه[7] هستند. به منظور حفظ ترتیب ورود کلمات در شبکه‌ی ترانسفورمر، ورودی شبکه با استفاده از ماسک پوشانده می‌شود تا شبکه نتواند ورودی بعد را مشاهده کند. با بررسی عملکرد مدل‌های مبتنی بر شبکه عصبی بازگشتی در وظیفه‌ی متن به پرس‌وجو بر روی محک‌هایی[8] مثل Spider مشاهده شدکه این مدل‌ها عملکرد بسیار ضعیفی روی این محک‌ها دارند. به همین علت است که عمدتا کارهای جدید بر پایه‌ی ترانسفورمر پیاده سازی می‌شوند. از این رو در مقالات [5 - 6]، [11] ,[13 - 15] از ترانسفورمر به عنوان هسته‌ی اصلی رمزنگار استفاده شد.

[1] Supervised
[2] Long Short-term Memory
[3] Encoder
[4] Transformer
[5] Vanishing Gradient
[6] Exploding Gradient
[7] Self-Attention
[8] Benchmark

۲-۴ جمع بندی

در جدول 1 به معرفی و مقایسه‌ی مدل‌های مختلفی که برای حل مسئله‌ی متن به پرس و جو ساخته شدند پرداختیم.

جدول 1مقایسه‌ی معماری و ویژگی‌های روش‌های مختلف برای حل مسئله‌ی تبدیل متن به پرس‌وجو

نام مدل	دیتاست	روش تولید پرس‌وجو	روش یادگیری		معماری	
			با ناظر	تقویتی	LSTM	TRANS.
SDSQL *[5]*	WikiSQL	مبتنی بر طرح	✓			✓
SQLova *[13]*	WikiSQL	مبتنی بر طرح*	✓			✓
X-SQL *[6]*	WikiSQL	مبتنی بر طرح	✓			✓
HydraNET *[14]*	WikiSQL	مبتنی بر طرح	✓			✓
Seq2SQL *[4]*	WikiSQL	مبتنی بر تولید		✓	✓	
SQLNet *[8]*	WikiSQL	پرکردن جای‌خالی	✓		✓	
SeqPolicyNet *[7]*	Movie Dialog	پرکردن جای‌خالی*		✓	✓	
DBPal *[9]*	GeoQuery	مبتنی بر تولید*	✓		✓	
NSP *[10]*	GeoQuery ATIS	مبتنی بر تولید*	✓		✓	
IRNet *[11]*	Spider	مبتنی بر تولید†	✓		✓	✓
BRIDGE *[15]*	Spider WikiSQL	مبتنی بر تولید†	✓			✓

*: از مکانیزم اشاره‌گر نیز بهره می‌برد.

†: مدل Sequence-to-tree

3- نوآوری‌های پژوهش:

قصد داریم در این پژوهش یک معماری جدید شبکه‌ی عصبی عمیق به منظور تبدیل متن به پرس‌وجوی SQL معرفی کنیم. همچنین دو مدل زبانی ELMo و BERT را برای پیدا کردن بهترین تعبیه در بخش رمزنگار، بررسی خواهیم کرد [16 - 17].

4- اهداف پژوهش:

۱) طراحی مدلی که بیشترین دقت را بر روی محک Spider [1] یا WikiSQL [2] داشته باشد نخستین هدف این پژوهش است.

۲) مقایسه عملکرد دو مدل زبانی BERT و ELMo بر روی وظیفه‌ی تبدیل متن به پرس‌وجو دومین هدف ما خواهد بود.

5- روش پژوهش:

مدلی که در این پژوهش تعریف می‌کنیم، دارای دو بخش رمزنگار و رمزگشا است. مانند سایر مدل‌های موجود ورودی ما توکن‌های سؤال کاربر، نام ستون‌های پایگاه‌داده و رابطه‌ی میان ستون‌ها خواهد بود. علاوه بر این‌ورودی‌ها، از مقادیر یکتای درون هر ستون پایگاه‌داده نیز به عنوان ورودی استفاده می‌کنیم. این موضوع به مدل کمک می‌کند که ارتباط میان کلمات درون سؤال کاربر و ستون پایگاه‌داده را بهتر یاد بگیرد [15]. در ادامه، در **بخش رمزنگار** از مدل زبانی ELMo یا BERT استفاده می‌کنیم. استفاده از مدل زبانی می‌تواند برای تعبیه‌ی همزمان ستون‌ها و مقادیر یکتای درون ستون‌ها، با توکن‌های سوال کاربر باشد [13]، و یا صرفا برای تعبیه‌ی توکن‌های سؤال کاربر باشد [14]. درصورتی که تعبیه‌ی ستون‌ها و مقادیر یکتایشان را جدای از مدل پیش‌آموز درنظر بگیریم، امکان استفاده از شبکه‌های عصبی مانند CNN[3] یا LSTM را برای رمزنگاری اطلاعات پایگاه‌داده بدست می‌آوریم. در **بخش رمزگشا**، از یک ماژول جهت انتخاب یک الگو برای تولید SQL بهره می‌بریم و تمام ترتیب خروجی مدل، بر اساس آن الگو خواهد بود. همچنین فضای خروجی رمزگشا را محدود می‌کنیم. به این معنا که خروجی مدل تنها توکن‌هایی را انتخاب می‌کند که در توکن‌های سؤال کاربر، توکن‌های دستورالعمل SQL، نام ستون‌های پایگاه و یا مقادیر یکتای هر ستون پایگاه باشد. ما در هر دوبخش رمزنگار و رمزگشا، تلاش می‌کنیم از معماری جدیدی استفاده کنیم تا بتوانیم یک نمایش بهتر از جمله، الگوی پایگاه‌داده و مقادیر یکتای ستون‌ها ارائه دهیم. در پایان به روش مقایسه‌ی نتایج اجرای دو پرس‌وجو[4] عملکرد مدل را می‌سنجیم.

در راستای رسیدن به اهداف پژوهش، معماری ما دچار چند چالش است که باید مورد بررسی قرار گیرد. در ادامه به ترتیب از ورودی تا خروجی به بررسی چالش‌های طراحی مدل می‌پردازیم.

اول) در بخش ورودی و تبدیل کلمات به بردار تفکر چند روش قابل استفاده است. روش نخست استفاده از تعبیه‌های ثابت[5] و دوم روش‌های پویا است. بنظر می‌رسد که استفاده از مدل‌های زبانی پیش‌آموز شده در افزایش دقت نهایی مدل موثر هستند [5 - 6] ، [11] و [13 - 14]. به این منظور در مدل پیشنهادی از مدل‌های زبانی پیش‌آموز شده مانند BERT و یا ELMo استفاده خواهد شد. با مقایسه‌ی ساختار دو مدل پیش‌آموز شده‌ی BERT و ELMo، استنباط می‌شود که استفاده از BERT در مقابل ELMo طراح را برای تغییرات اساسی در بخش رمزنگار محدود می‌کند. این امر درمقابل ELMo که صرفا به عنوان ورودی یک جمله دریافت و در خروجی تعبیه‌ی هر کلمه را ارائه می‌دهد متفاوت است. یکی از اهداف ما پیدا کردن بهترین مدل زبانی پیش‌آموز شده برای وظیفه‌ی متن به پرس‌وجو است.

[1] https://yale-lily.github.io/spider
[2] https://github.com/salesforce/wikisql
[3] Convolutional Neural Network
[4] Query Result Comparison
[5] Static Embeddings: Glove, Word2vec

دوم) گام دوم انتخاب نحوه‌ی ورود اطلاعات به مدل است. همانطور که پیش‌تر گفته شد، ورودی از دو بخش سؤال کاربر به زبان طبیعی و شمای پایگاه‌داده تشکیل می‌شود. در این مرحله چالش ما این است که ورودی را به نحوی وارد مدل کنیم که مدل توانایی درک و یادگیری ارتباطات درونی پایگاه و همچنین ارتباط پایگاه‌داده با کلمات درون جمله‌ی سوال را به خوبی درک کند. مراجع [5 - 6] و [13 - 14] از BERT برای تعبیه کلمات استفاده کردند. مراجع [5 - 6] و [13] ستون‌های پایگاه را به همراه طرح کلی پایگاه به عنوان ورودی وارد BERT کردند. در مقابل این روش، مرجع [14] یک ساختار متفاوت برای ترکیب ستون و ارتباطات درونی پایگاه‌داده با متن سوال کاربر ارائه کرد.

سوم) انتخاب الگوی مناسب برای ساخت ترتیب خروجی نیز یکی دیگر از چالش‌های طراحی مدل است. از آنجا که انتخاب الگو مرحله‌ی نخست ساخت ساخت ترتیب خروجی‌است، این مرحله به عنوان نوعی گلوگاه عمل می‌کند و درصورت انتخاب اشتباه الگو، کل ترتیب خروجی اشتباه خواهد بود.

چهارم) بررسی صحت عملکرد مدل خود به سه روش صورت می‌گیرد. مقایسه نتایج، تطبیق درخت تجزیه و تطابق رشته‌ی متن[1]. تطابق رشته‌ی متن دقت را از حد واقعی کمتر نمایش‌می‌دهد، درخت تجزیه خطای کمتری در بررسی عملکرد دارد و مقایسه‌ی نتایج ممکن است پرس‌وجوی نادرست را درست تشخیص دهد. با این وجود روش ترجیح داده شده توسط دیتاست Spider مقایسه نتایج است.

9- فهرست مراجع اصلی:

[1] Google inc., "Google Begginer SEO Documentation," [Online]. Available: https://developers.google.com/search/docs/beginner/how-search-works.

[2] J. Chapuis, "Natural Language Interfaces to Databases (NLIDB)," 2017. [Online]. Available: https://www.nexthink.com/blog/natural-language-interfaces-databases-nlidb.

[3] K. Ahkouk and M. Machkour, "Towards an interface for translating natural language questions to SQL: a conceptual framework from a systematic review," *International Journal of Reasoning-based Intelligent Systems*, pp. 264-275, 2020.

[4] V. Zhong, C. Xiong and R. Socher, "Seq2sql: Generating structured queries from natural language using reinforcement learning," *arXiv preprint arXiv:1709.00103*, 2017.

[5] B. Hui, X. Shi, R. Geng, B. Li, Y. Li and J. Sun, "Improving Text-to-SQL with Schema Dependency Learning," *arXiv:2103.04399v1*, 2021.

[6] P. He, Y. Mao, K. Chakrabarti and W. Chen, "X-SQL: reinforce schema representation with context," *arXiv:1908.08113*, 2019.

[7] S. Blank, F. Wilhelm, H. Zorn and A. Rettinger, "Querying NoSQL with Deep Learning to Answer Natural Language Questions," in *IAAI*, 2019.

[8] X. Xu, C. Liu and D. Song, "Sqlnet: Generating structured queries from natural language without reinforcement learning," *arXiv:1711.04436*, 2017.

[9] F. Basik, B. Hättasch, A. Ilkhechi, A. Usta, S. Ramaswamy, P. Utama, N. Weir, C. Binnig and U. Cetintemel, "Dbpal: A learned nl-interface for databases," in *International Conference on Management of Data*, 2018.

[10] S. Iyery, I. Konstasy, A. Cheung, J. Krishnamurthyz and L. Zettlemoyer, "Learning a neural semantic parser from user feedback," *ACL*, p. 963–973, 2017.

[11] J. Guo, Z. Zhan, Y. Gao, Y. Xiao and J. Lou, "Towards Complex Text-to-SQL in Cross-Domain Database with Intermediate Representation," *ACL*, p. 4524–4535, 2019.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser and I. Polosukhin, "Attention is all you need," *CoRR*, 2017.

[1] Query String Matching

[13] W. Hwang, J. Yim, S. Park and M. Seo, "A Comprehensive Exploration on WikiSQL with Table-Aware Word Contextualization," *arXiv:1909.04165*, p. 2019, 2019.

[14] Q. Lyu, K. Chakrabarti, S. Hathi, S. Kundu, J. Zhang and Z. Chen, "Hybrid Ranking Network for Text-to-SQL," *arXiv:2008.04759v1*, 2020.

[15] X. V. Lin, R. Socher and C. Xiong, "Bridging textual and tabular data for cross-domain text-to-sql semantic parsing," *arXiv preprint arXiv:2012.12627*, 2020.

[16] J. Devlin, M. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv:1810.04805*, 2018.

[17] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee and L. Zettlemoyer, "Deep contextualized word representations," in *NAACL*, 2018.