
بسم الله الرحمن الرحيم

دانشگاه اصفهان

دانشکده مهندسی کامپیوتر

گروه مهندسی فناوری اطلاعات

گزارش پروژه کارشناسی

رشته مهندسی کامپیوتر گرایش فناوری اطلاعات

عنوان پروژه

طراحی و پیاده سازی برنامه‌ای برای تشخیص اشتباهات املایی زبان فارسیِ زبان آموزان غیر بومی توسط شبکه عصبی

استاد راهنما:

دکتر مرجان کائدی

پژوهشگران:

هومان هنرور

مروارید رهبر

شهریور ۱۴۰۳

دانشگاه اصفهان

دانشکده مهندسی کامپیوتر

گروه مهندسی فناوری اطلاعات

پروژه کارشناسی رشته‌ی مهندسی کامپیوتر گرایش فناوری اطلاعات

آقای/خانم .....

تحت عنوان

.....

در تاریخ / / ۱۳ توسط هیأت داوران زیر بررسی و با نمره به تصویب نهایی رسید.

۱- استاد راهنمای پروژه:

دکتر امضا

۲- استاد داور :

دکتر امضا

امضای مدیر گروه

تشکر و قدردانی

این صفحه اختیاری است و می‌توانید از کسانی که شما را در انجام این پروژه یاری رسانده‌اند تشکر و قدردانی نمایید.

چکیده:

هنگامی که یک زبان‌آموز، در حال یادگیری یک زبان جدید به غیر از زبان مادری خود است، در این مسیر یادگیری، اشتباهات و خطاها، جزئی جدایی ناپذیر و مهم در مسیر پیشرفت و یادگیری او هستند. خطاهای دستوری و نگارشی بخش عمده‌ای از انواع اشتباهات بین زبان‌آموزان، را تشکیل می‌دهند. این خطاها را با توجه به حوزه هایی که در آن رخ می‌دهند می‌توان در دسته بندی های متفاوتی قرارداد. برای مثال، خطاهای نوشتاری به پنج حوزه‌ی املایی، دستوری، صرفی، معنای و سبکی تقسیم می‌شوند. در این پروژه بر روی خطاهای املایی تمرکز می‌شود.

تصحیح غلط‌های املایی یک مساله مورد توجه در زمینه پردازش زبان طبیعی است. هدف از این پروژه، طراحی و پیاده‌سازی برنامه‌ای است که اشتباهات املایی زبان‌آموزان زبان فارسی را به صورت خودکار شناسایی کند و بدین ترتیب، در فرآیند آموزش زبان فارسی به آنها کمک کند و آموزش زبان فارسی را برای آنها تسهیل کند. این تشخیص، با توجه به بستر متنی که کلمات در آن واقع هستند و ایجاد یک طبقه بندی به منظور شناخت غلط‌های املایی محتمل برای زبان‌آموزان با در نظر گرفتن ملیت آنها است.

برای انجام این پروژه، نیاز به یک مجموعه داده برچسب‌دار است. این مجموعه داده، حاوی متونی است که توسط زبان‌آموزان زبان فارسی با ملیت‌های مختلف، نوشته شده‌باشند و سپس غلط‌های املایی آنها برچسب گذاری شده‌باشد. چنین مجموعه داده‌ای در حال حاضر در گروه زبان شناسی دانشگاه اصفهان فراهم شده‌است و در دسترس است. سپس مدل‌های شبکه عصبی بر روی این مجموعه داده آموزش داده شد که بتوانند برچسب‌گذاری غلط‌های املایی را به صورت خودکار انجام دهند. پس از ارزیابی و بهبود مدل، برنامه کاربردی‌ای به صورت تحت وب توسعه داده‌شد و این مدل در آن برنامه تعبیه شد تا زبان‌آموزان بتوانند در جهت تسهیل و بهبود فرآیند آموزش خود از آن استفاده کنند.

واژگان کلیدی: پردازش زبان طبیعی، اشتباهات املایی، زبان فارسی، زبان‌آموزان، دسته‌بندی، شبکه عصبی

فهرست مطالب

عنوان **صفحه**

فهرست مطالب

عنوان **صفحه**

فهرست مطالب

عنوان **صفحه**

فهرست شکل‌ها

عنوان **صفحه**

فهرست جدول‌ها

عنوان **صفحه**

مخففها:

XMI	XML Metadata Interchange
Bleu	Bilingual Evaluation Understudy
Mn	Main Signs
Dc	Diacrictic Signs
F	Form
Pu	Punctuation
C	Consonant
Vl	Vowel
H	Hamza
L	Long Vowels
Sh	Short Vowels
Tan	Tanwin
Ta	Tashdid
Ma	Mad
B	Boundary
Sp	Separation
Mg	Merging
Da	Dandaneh
Do	Dot
Pl	Place
UML	Unified Modeling Language
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
GRU	Gated recurrent unit
NLP	Natural language processing
TP	True Posetive
TN	True Negative
FP	False Posetive
FN	False Negative
BP	Brevity Penalty
Wn	weight for n-gram
MVT	Model View Template
MVC	Model View Controller
HTTP	Hypertext Transfer Protocol
HTML	Hypertext Markup Language
T-SNE	T-distributed Stochastic Neighbor Embedding
TF-IDF	Term Frequency - Inverse Document Frequency
BOW	Bag of Words
PAD	Padding Token
MT5	Multilingual T5
XLM-R	XLM-RoBERTa
RoBERTa	Robustly Optimized BERT Approach

BPE	Byte Pair Encoding
SOS	Start of Sentence
EOS	End of Sentence
UNK	Unkown Token
DFF	Dimension Feed Forward

فصل اول
مقدمه

۱-۱- هدف پروژه

هدف از انجام این پروژه پیدا کردن روابطی بین ملیت زبان‌آموزان زبان فارسی و نوع غلط املائی‌هایی که مرتکب می‌شوند، می‌باشد. میسر شدن چنین هدفی در مرحله نخست مستلزم درک و شناخت دقیقی از انواع خطاها و در نتیجه بدست آوردن یک مجموعه داده از داده‌های غلطی که توسط زبان‌آموزان نوشته شده و داده‌های صحیح شده آنها و برچسب گذاری آن داده ها است. این مجموعه داده توسط گروه زبان‌شناسی دانشگاه اصفهان تهیه شده و در اختیار این پروژه قرار داده شده ، در نهایت پروژه انجام شده و در بستر تحت وب پیاده‌سازی آماده استفاده برای کاربران شده‌است.

۱-۲- کاربردهای پروژه

این پروژه با شناخت و درک دقیق از ملیت زبان‌آموزان و غلط‌های املایی مکرر آنها می‌تواند محیط مناسب‌تری برای آموزش زبان فارسی برای آنها باشد. زبان‌آموزان با ملیت های گوناگون که علاقه‌مند به یادگیری زبان فارسی هستند، می‌توانند با وارد کردن ملیت خود در کنار جملات ورودی خود از مدل هوش مصنوعی انتظار تصحیح خطای بهتری داشته‌باشند و مراحل یادگیری زبان فارسی را برای آنها هموار تر سازد.

۱-۳- ارزش پروژه

این پروژه در مرحله اول به عنوان یک تحقیق و آزمایش خطا و در ادامه با نتیجه دادن و تحقق یافتن هدف، یک بستر مناسب برای یادگیری زبان فارسی معرفی می‌شود. ما به دنبال این هستیم که آیا ریشه زبان مادری یکسان، آیا نوع درک یکسانی از یادگیری زبان جدید متعاقبا به همراه می‌آورد، یا خیر.

۴-۱- ساختار پایان نامه

در این پروژه به دنبال پیدا کردن جواب منطقی و قابل استناد لازم است ابتدا با انواع خطاهایی که توسط مصححان شناسایی شده‌اند آشنا شویم. سپس باید استخراج سازی داده‌ها از فایل های دریافتی از دانشکده زبان شناسی انجام شود. از آنجایی که فرمت فایل ها در قالب XMI[1] است لازم است یک آشنایی با این فایل‌ها صورت گیرد و ویژگی‌هایی که در این فایل‌ها تعریف شده‌اند را بشناسیم. از آنجایی که این فایل‌های دریافتی از دانشکده زبان شناسی خروجی از برنامه INCEPTION است یک آشنایی با کارکرد این برنامه، به درک بهتر فایل‌های خروجی کمک می‌کند. بعد از استخراج سازی داده‌های ارزشمند به صورت یک فایل اکسل منسجم، بر روی داده‌ها یک اندازه گیری و آمارگیری انجام می‌دهیم برای اطلاعات دقیق تر از تعداد خطاها در نوع و ملیت‌های متفاوت.

با داده بدست آمده، ابتدا کاهش بعد و حذف داده پرت انجام می‌دهیم، سپس چند دسته‌بندی (Clustering) با الگوریتم های متفاوت انجام می‌دهیم که یک دید کلی داشته باشیم که آیا داده‌هایی که در دسترس هستند قابل جداسازی و آموزش به عنوان ملیت های جدای از یکدیگر هستند یا خیر (این قسمت یک دید بهتر در نتیجه گیری و استنباط به ما می‌دهد).

در ادامه ابتدا یک مدل شبکه عصبی مناسب انتخاب کرده و متناسب با آن مدل شبکه عصبی پیش پردازش‌های مربوطه را انجام شده‌است (البته پیش پردازش ها نیز باید با وظیفه تصحیح خطا نیز همخوانی داشته باشند). پس از بدست آوردن مدل ایده‌آل و آموزش مدل توسط داده‌ها، مدل آموزش دیده توسط داده‌های تست از قبل جدا شده ارزیابی می‌شود و در چهار معیار Bleu_Score[2], f1-score,Recall,Precision Accuracy, بیان و تحلیل می‌شوند.

مدل‌های نهایی را به دو قسمت تقسیم می‌کنیم: آموزش با برچسب ملیت و آموزش بدون برچسب ملیت، سپس نتیجه تست هرکدام از مدل‌ها را با یکدیگر مقایسه می‌کنیم که آیا برچسب ملیت تاثیر مثبت و به سزایی در فرایند یادگیری داشته است؟ بدین معنی که مدل با دانستن ملیت مربوطه نویسنده غلط های رایج تر که در زبان‌آموزان آن کشور را بهتر درک کرده‌است ؟

در نهایت برای استفاده بهتر و راحت تر از مدل‌های شبکه عصبی، یک برنامه تحت وب پیاده‌سازی شده‌است که با گرفتن جمله ورودی از کاربر و گرفتن نوع مدل یا ملیت نویسنده خروجی را برای کاربر به نمایش می‌گذارد.

---

[1] XML Metadata Interchange
[2] Bilingual Evaluation Understudy

فصل دوم
مفاهیم

۲-۱- مقدمه

پردازش زبان طبیعی انسان یکی از شاخه‌های کاربردی هوش مصنوعی است که توجه بسیاری را به سمت خود جلب کرده‌است. این رشته شامل زیر رشته های بسیاری میباشد که یکی از آنها تصحیح خطا میباشد. مدل‌ها و استراتژی‌های گوناگونی برای عملی کردن این عملیات طراحی و پیاده‌سازی شده‌است. ابتدا برای درک بهتر از این زیر رشته در زبان فارسی بهتر است با انواع خطاها در زبان فارسی آشنا شویم در ادامه نگاهی گذرا به نحوه استخراج سازی و برچسب گذاری برای تولید و تهیه داده‌ها که توسط برنامه INCEPTION انجام می‌شود، داریم و سپس به بررسی و شرح مناسب و کامل مدل‌هایی که قادر به پردازش این کار هستند پرداخته می‌شود، از آنجایی که مدل‌های نهایی در بستر وب تعبیه و استفاده می‌شوند، نگاهی گذرا به معماری و ساختار برنامه تحت وب نیز داریم.

۲-۲- انواع خطا

هر زبانی دارای نظام نگارشی خاصّ خود است که شامل مجموعه‌ای از اصول و قواعد قراردادی است. این اصول ناظر بر چگونگی نگارش حروف، قواعد چینش و ترتیب آن‌ها می‌باشد. در یک تعریف کلّی، املاء شیوه‌ی صحیح نگارش یک زبان خاص است توسط حروف الفبای تشکیل دهنده‌ی آن زبان که اگر انحرافی از صورت استاندارد اصول نگارشی آن‌ها صورت گیرد به آن خطای املائی می‌گویند. به بیانی دقیق‌تر خطاهای املائی، خطاهای شناختی‌ای هستند که از صورت صحیح نگارشی منحرف شده و جایگزین املای درست کلمه می‌شوند. این خطاها معمولاً در نتیجه‌ی عدم آگاهی نویسنده از املای صحیح یک کلمه، فراموش کردن صورت نوشتاری صحیح یا درک نادرست از شیوه‌ی مناسب نگارش کلمه رخ می‌دهند.

با توجّه به تعاریفی که از خطاهای املائی ارائه شـد، در حوزه‌ی املایی خطاهای زیر مشـاهده و به همراه برچسـب مخصـوص به هر خطا دسـته‌بندی و نام‌گذاری شـدند. برای برچسـب خطاهای املایی به عنوان یک حوزه‌ی کلّی برمبنای معادل لاتین آن یعنی واژه‌ی (Orthography) به اختصار کد (Or) در نظر گرفته شده است. در سطح مقوله ابتدا خطاهای املائی را به چهار مقوله‌ی اصلی خطاهای نشانه‌ی اصلی(Mn^1)، خطاهای نشانه‌ی ثانوی(Dc^2)، خطاهای فرم (F^3) و خطای علائم نگارشی (Pu^4) تقسیم‌بندی نمودیم و سپس برای هر یک از مقولات، زیرمقولاتی درنظر گرفته شـد که به وسـیله‌ی یک خط تیره و با حروفی کوچک به مقولات اصـلی متّصـل می‌شـوند. تقسـیم‌بندی مقولات خطایی حوزه‌ی املائی به دو دسـته‌ی خطاهای نشـانه‌ی اصـلی و خطاهای نشـانه‌ی ثانوی را بر اسـاس تقسـیم‌بندی دسـتور مصـوّب فرهنگسـتان برگزیدیم و سـطح سـومی بنام خطای فرم یا صورت را برای آن دسته از خطاهای صوری که منجر به تولید صورت نابه هنجار کلمه می‌شوند در کنار دو مقوله‌ی اصـــلی قبل اضـــافه نمودیم. در نهایت خطاهای مربوط به علائم نگارشـــی را نیز به عنوان زیرمجموعه‌ای برای خطای املائی قرار داده و به عنوان مقوله‌ی چهارم در نظر گرفتیم. در ســـطح آخر نیز فرآیندهای خطایی مناسب با هر مقوله به کدهای قبلی افزوده شده‌اند.

۱-۲-۲- خطاهای نشانه اصلی

اوّلین گروه از خطاهای املایی، خطاهای نشانه‌ی اصلی هستند. نشانه‌های اصلی در زبان فارسی در واقع همان حروف الفبای فارســـی هســـتند که به عنوان معادل واکه‌ها و همخوان‌های زبان فارســـی قرار می‌گیرند. بنابراین در این نوع خطا در سـطح دوم کدهای خطایی هم شـاهد وجود یک مقوله‌ی اصـلی هسـتیم و هم دو زیرمقوله‌ی آن یعنی خطاهای همخوان بنیاد[5] و خطاهای واکه بنیاد[6] که توسـط فرآیندهای خطایی گوناگون از یکدیگر متمایز می‌شوند.

۱-۱-۲-۲- خطاهای همخوان بنیاد

به خطاهایی که بر روی همخوان‌های زبان فارسـی در حوزه‌ی املائی صـورت می‌گیرد خطاهای همخوان بنیاد می‌گویند چراکه اسـاس خطای رخ داده همخوان‌ها می‌باشـند و نه واکه‌ها. در رابطه با خطاهای همخوانی هر چهار نوع فرآیند خطایی مشاهده شده است. به این ترتیب پس از کد (Or) که معرّف حوزه‌ی املائی است می‌بایسـت کد مربوط به مقوله‌ی خطایی وارد شـود. در اینجا علاوه بر مقوله‌ی خطایی نشـانه‌ی اصـلی که به وســـیله‌ی کد (Mn) نمایش داده می‌شـــود، زیر مقوله‌ی (C^7) مبیّن گروه همخوان‌ها با حرف کوچک و به

---

[1] Main Signs
[2] Diacrictic Signs
[3] Form
[4] Punctuation
[5] Consonant
[6] Vowel
[7] Consonant

وسیله‌ی خطّ تیره‌ای به مقوله‌ی اصلی (Mn) می‌چسبد. بنابراین تا بدین جای کار کد (Or,Mn-c) در دو سطح ساخته شده است. در ادامه به تفکیک فرآیندهای خطایی رخ داده شده، سایر کدهای این مقوله معرفی شده‌اند.

الف) خطای جابه‌جایی نویسه‌ی همخوانی در کلمه

همانطور که در یک جمله ممکن است ترتیب کلمات به‌هم ریزد و چینش آن‌ها فرم ناصحیحی به خود گیرد، در یک کلمه نیز همین خطا ممکن است برای حروف تشکیل دهنده‌ی آن کلمه رخ دهد. در این نوع خطا عنصری حذف یا اضافه نمی‌شود بلکه تمام عناصر حضور دارند و تنها جایگاه مناسب خود را تغییر داده و در جایی غیر از جایگاه صحیح خود قرار گرفته‌اند. مانند ترتیب ناصحیح حروف در کلماتی چون مشکیل، پیوران، پبسند، سرماییه در ازای تولید واژه‌های مشکلی، پیروان، بپسند و سرمایه‌ی. برچسب مربوط به فرآیند جابه‌جایی کد (O) است که در کنار کد (Or,Mn-c) برچسب (Or,Mn-c,O) را می‌سازد. در نمونه‌های زیر این خطا قابل مشاهده است. در هر یک از نمونه‌ها جهت حفظ اصالت داده، تمامی خطاهای صورت گرفته در جمله عیناً ثبت شده‌اند و جهت اجتناب از ابهام زیر خطای مورد نظر خط کشیده شده است.

نمونه:

۱) شهر بمبئی شهر زیبا و برزگ است.

< Or,Mn-c,O >برزگ\بزرگ< Or,Mn-c,O />

۲) در شهر مهشد حرم امام رضا است.

< Or,Mn-c,O >برزگ\بزرگ< Or,Mn-c,O />

۳) آنان برای امامت بردیگزه شدند.

< Or,Mn-c,O >بردیگزه\برگزیده< Or,Mn-c,O />

در شرایطی که این خطا همزمان با خطای دیگری در کلمه رخ دهد هر دو کد به وسیله‌ی یک خط مورّب به عنوان تحلیل‌های متفاوت یک خطا در کنار یکدیگر قرار می‌گیرند و به طریق زیر برچسب‌گذاری می‌شوند:

۴) جمیعت هید خیلی زیاد است.

<Or,Mn-c,O /Or,Dr-ta,M >جمیعت\جمعیّت< Or,Mn-c,O / Or,Dr-ta,M />

در نمونه‌ی بالا از یک سو ترتیب نگارش و چینش حروف در یک کلمه رعایت نشده است و از سوی دیگر نشانه‌ی ثانوی تشدید حذف شده است و از آنجا که نمیتوان تاخیر و تقدّمی قطعی برای فرآیندهای خطایی صورت گرفته در نظر گرفت، بنابراین هر دو کد خطایی به وسیله‌ی خطی مورّب در کنار یکدیگر برچسب‌گذاری شده‌اند تا نشان داده شود دو خطا به صورت همزمان در کلمه رخ داده‌اند.

نکته‌ای که در همین‌جا لازم است به آن اشاره شود، اشاره به چگونگی ساختار و الگوی برچسب‌دهی خطاهای فوق است. الگوی اصلی این شیوه از برچسب‌دهی برگرفته از نظام برچسب‌زنی پیکره‌ی سی. ال. سی(2012) می‌باشد. الگوی کلّی به شرح زیر است (نیکولز، ۲۰۰۳):

<کد#|>کلمه‌ی صحیح\کلمه‌ی ناصحیح<کد#>

همانطور که از الگو پیدا است در آغاز و پایان الگو، برچسب مربوط به خطا آورده می‌شود. علّت تکرار کد به این دلیل است که بتوان در رابطه با خطاهایی که از دو جنبه قابل تحلیل هستند، کد دوم را به عنوان کدی در درون کد اصلی نمایش داد به گونه‌ای که کد مربوط به خطای مهم‌تر و اوّلیه در دو سمت بیرونی الگو قرار بگیرند و کد دوم درون الگو. پس از نگارش کد، کلمه با صورت خطامند آن آورده می‌شود و بلافاصله بعد از آن صورت اصلاح‌شده و صحیح همان کلمه نیز نوشته می‌شود. نکته‌ای که در اینجا حائز اهمیّت است این است که خروجی هر برچسب جهت نشان‌گذاری، بسته به اطّلاعاتی که به عنوان تنظیمات وارد بدنه‌ی اصلی پیکره می‌شود، متفاوت است. بنابراین تا زمانی که این برچسب‌ها به صورت دقیق وارد پیکره‌ای نشوند و به عنوان کدهایی جهت نشان‌گذاری پیکره تعریف نگردند نمی‌توان با اطمینان از خروجی قطعی‌ای صحبت نمود. به دلیل گفته شده و با اطمینان از اینکه تغییراتی در خروجی الگوی برچسب‌دهی حاصل خواهد شد، در این پژوهش صرفاً چارچوبی کلّی را الگوبرداری نموده‌ایم تا بتوان برچسب‌دهی نمونه‌ها را به صورت عینی‌تر نمایش داد. در این راستا تغییراتی نیز بر الگوی فوق در شرایط وجود چند خطا یا امکان تحلیل خطا از دو بعد اعمال شد. حال آنکه آنچه در ارائه‌ی برچسب‌ها واضح است این است که محقّق تنها با هدف شناسایی، دسته‌بندی و عنوان‌دهی خطاها کدهایی را به عنوان پیشنهاد ارائه می‌دهد که این امر براساس دانش زبانشناسی صورت می‌گیرد با این وجود تا حدّ زیادی هم سلیقه‌ای است. بنابراین کاربرانی که قصد استفاده از این مجموعه برچسب‌ها را دارند می‌توانند بسته به هدف خود آن‌ها را تغییر داده و تعدیل نمایند.

ب) خطای افزایش نویسه‌ی همخوانی

در این خطا حرف یا حروفی در جایی که ضرورتی ندارد به کلمه افزوده می‌شوند. مانند کلمات خطامندی چون اعام الفیل، گاندهی، گره‌سنه، منمبر و غیره به جای تولید واژه‌های درست عام‌الفیل، گاندی، گرسنه و منبر. این خطا در پیکره توسط کد (Or,Mn-c,A) نشان‌گذاری می‌گردد.

ج) خطای حذف نویسه‌ی همخوانی

حذف حرف یا حروفی در یک کلمه یکی از عواملی است که موجب تولید خطا می‌شود. به عنوان مثال حذف حروف در کلماتی چون مشورترین و آخری به جای مشهورترین و آخرین که در مورد اوّل حرف (ه) و در مورد دوم حرف (ن) حذف شده است موجب تولید کلماتی شده که به لحاظ املایی خطامندند. این خطا در پیکره به همراه کد (Or,Mn-c,M) نمایش داده می‌شود.

خطای حذف حروف می‌تواند شامل حذف بیش از یک حرف هم شود مانند حذف دو حرف (میم) و (

الف) در کلمه‌ی می‌فرماید که موجب تولید کلمه‌ی می‌فرید می‌گردد.

د) خطای جایگزینی نویسه‌ی همخوانی

در این نوع خطا حرف یا حروفی جایگزین نویسه‌ی همخوانی دیگر می‌شود. به عنوان مثال در کلمات رزک، ابادت و مشقل به ترتیب حروف (ک، ا و ق) جایگزین حروف (ق، ع و ک) شده‌اند. از آنجا که این نوع جایگزینی یکی از ساده‌ترین انواع جایگزینی حروف است تنها توسط کد (Or,Mn-c,R) نشان‌گذاری می‌گردد.

با دقّت نظر بیشتری در نمونه خطای جایگزینی حروف می‌توان دریافت که جایگزینی در حروف می‌تواند به سبب وجود عامل‌های مختلف دیگری نیز باشد؛ به عنوان مثال جایگزینی یک حرف با حرف دیگر در نتیجه‌ی افزودن یا کاهش نقطه، دندانه، سرکش و غیره نمونه‌هایی از این عوامل هستند. با اینکه این موارد را در مجموعه برچسب حاضر به عنوان خطاهای فرم در نظر گرفته‌ایم امّا در اینجا چنین فاکتورهایی را نمی‌توان به عنوان خطای فرم محسوب نمود چراکه در تعریف خطاهای فرم خواهیم دید که خطاهایی در این مقوله می‌گنجند که موجب تولید ساختارهای نابه‌هنجار در زبان فارسی شوند؛ حال آنکه در خطای جایگزینی حروف، با حذف یا افزودن این فاکتورها حروفی تولید می‌شود که در زبان فارسی به عنوان یک همخوان وجود خارجی داشته و به کار می‌روند. به عنوان مثال حذف نقطه در کلمه‌ی شیراز (شیرار) یا افزودن سرکج به کلمه‌ی کر (گر) از جمله عواملی هستند که موجب جایگزینی دو همخوان با یکدیگر شده‌اند. بنابراین از آنجا که جایگزینی دو همخوان موجود در زبان فارسی با یکدیگر صورت می‌گیرد در زمره‌ی خطای همخوان‌بنیاد مورد بررسی قرار خواهند گرفت. هرچند ممکن است که عامل‌های صوری باعث ایجاد چنین جایگزینی‌ای شده باشند.

جایگزینی حروف ممکن است در نویسه‌گونه‌های متعلّق به یک حرف نیز رخ دهد. به عنوان نمونه حرف (الف) در مثال زیر جایگزین حرف (ی) شده است که نمود دیگری از همان حرف (الف) است امّا به دلیل مغایرت با صورت نوشتاری صحیح آن، خطا تلّقی می‌شود.

از این دست‌اند نویسه‌گونه‌هایی چون حرف (ت،ط)، (ض،ظ،ز،ذ) و غیره. از آنجا که بحث همخوان‌ها در خطای حاضر مطرح است و یکی از رایج‌ترین خطاهای زبان‌آموزان عدم توانایی در تشخیص و به کار بردن همخوان مورد نظر در نویسه‌گونه‌ها است و بالاخصّ اینکه در بسیاری از موارد این چندگانگی موجب تغییر در معنای کلمه‌ی مورد نظر می‌شود (ثواب، صواب)، در صدد برآمدیم تا به گونه‌ای این تمایز را در برچسب‌ها به نمایش بگذاریم. لذا در سطح دوم مقوله‌ی خطاهای نشانه‌ی اصلی و در کنار زیر مقوله‌ی خطاهای همخوان بنیاد یک معادل لاتین برای هر گروه برگزیدیم. به عنوان مثال برای همخوان چند نویسه‌ای (ت، ط) از معادل لاتین (T) و برای همخوان (ض، ز، ذ، ظ، ذ) معادل (Z) را انتخاب نمودیم که به وسیله‌ی یک خط تیره به زیرمقوله‌ی قبلی خود متّصل می‌شود. به این ترتیب در سطح مقوله، طی افزودن دو زیرمقوله به نمایش جزئیّات بیشتری از خطای مربوطه پرداخته‌ایم. در زیر هر یک از همخوان‌های چند نویسه‌ای به همراه کد در

نظر گرفته شده برای آن‌ها آورده شده است:

۲-۱-۲-۲- خطاهای واکه بنیاد

در زبان فارسی سه مصوت /æ/، /e/ و /o/ مصوّت‌هایی کوتاه هستند که در مقابل هر یک به ترتیب جفت کشیده‌ی آن‌ها یعنی مصوّت‌های /a/، /i/ و /u/ قرار می‌گیرد. در داده‌های بررسی شده موارد بسیاری از جایگزینی هر یک از این دو گونه با یکدیگر وجود دارد که تولید ساخت‌هایی خطامند را در پی داشته است. بنابراین برای این نوع خطا دو دسته‌بندی جزئی وجود دارد؛ الف) کوتاه شدن مصوّت‌های کشیده ب) کشیده شدن مصوّت‌های کوتاه. بدین ترتیب همانند خطاهای همخوان بنیاد، در سطح مقوله شاهد وجود دو زیرمقوله هستیم. بنابراین اگر حرف (Vl[1]) را معادل زیرمقوله‌ی خطاهای واکه بنیاد درنظر بگیریم، کد (Or,Mn-vl) تولید خواهد شد. درباره‌ی زیر مقوله‌ی دوم نیز درصورتی که حرف (l[2]) را نماینده واکه‌های کوتاهی بدانیم که به صورت کشیده آورده شده‌اند، بنابراین خواهیم داشت (Or,Mn-vl-l). همچنین در مقابل اگر حرف (sh[3]) را به عنوان کدی برای واکه‌های بلندی تعریف کنیم که به صورت کوتاه مورد استفاده قرار گرفته‌لند کد (Or,Mn-vl-sh) تولید خواهد شد. از آنجا که در رابطه با واکه‌ها، فقط خطای جایگزینی واکه‌ها مشاهده شده است بنابراین در این گروه تنها دو کد (Or,Mn-vl-sh,R) و (Or,Mn-vl-l,R) وجود دارد. همچنین حذف واکه می‌تواند در مورد حذف یک حرف در نویسه‌های چند حرفی معادل یک آوا نیز در نظر گرفته شود. مانند حذف حرف (واو) که بخشی از واجگونه‌ی (الف) مستثنی است یعنی (وا) در کلمه‌ی نخواندین که زبان‌آموز آن را به صورت نخاندین تولید نموده است. به علاوه اگر هر دو حرف واو و الف (وا) نیز به همراه هم حذف شوند از آنجا که هر دو باهم نویسه‌گونه‌ی معادل یک واکه هستند، مانند سایر خطاها با کد مربوط به همین مقوله برچسب‌گذاری می‌شوند.

۲-۲-۲- خطای نشانه‌ی ثانوی

در دسته‌بندی خطاهای املایی دیدیم که اوّلین دسته متعلّق به گروه خطاهای نشانه‌ی اصلی بود که شامل نویسه‌های معادل واکه‌ها و همخوان‌های زبان فارسی می‌شد. در بالا و پایین این حروف اصلی در رسم‌الخطّ فارسی نشانه‌هایی به کار می‌روند که در ایجاد تمایز و تلفّظ صحیح کلمه نقش ایفا می‌کنند. گرچه این نشانه‌ها فرع بر الفبای فارسی‌اند و در نگارش یا عدم نگارش آن‌ها در برخی موارد اختلاف نظرهایی وجود دارد امّا از آنجا که در زبان فارسی کاربرد داشته و نیاز است تا فارسی‌آموزان با انواع صورت‌های آن‌ها آشنایی پیدا کنند و عدم رعایت برخی از آنها طبق فرهنگ مصوّب فرهنگستان به تولید صورت‌های خطامند منجر می

[1] Vowels
[2] Long Vowels
[3] Short Vowels

شود، در این مجموعه برچسب به آن‌ها اشاره شده است. همچنین علّت دیگر انتخاب نشانه‌های ثانوی به دلیل کم اهمیّت بودن این خطاها نسبت به خطاهای اصلی است. نشانه‌هایی چون تشدید، تنوین، مد، الف مقصوره، یاء کسره‌ی اضافه در پایان کلمات مختوم به های غیر ملفوظ و غیره از این نوعند. کد مربوط به خطای نشانه‌های ثانوی برگرفته از سرواژه‌ی معادل لاتین آن حرف (Dc) انتخاب شده است. در ادامه زیر مقولات مربوط به این خطا آورده شده است.

۱-۲-۲-۲- خطای تنوین

تنوین، یکی از نشانه‌های زبان عربی است که وارد زبان فارسی شده و به عنوان نشانه‌ای ثانویه مورد استفاده قرار می‌گیرد. این نشانه دارای سه گونه‌ی تنوین نصب(اً)، تنوین ضم (اٌ) و تنوین جر (اٍ) می‌باشد؛ امّا در زبان فارسی آنچه رواج بیشتری دارد همان مورد اوّل، یعنی تنوین نصب است که همراه با کرسی (الف) در پایان برخی کلمات آورده می‌شود. با توجّه به این‌که تنوین نشانه‌ای مازاد بر حروف به کار رفته در یک کلمه است، در برخی موارد از نگارش آن امتناع می‌شود و دلیل آن هم این است که در زبان فارسی اصراری بر نگارش نشانه‌های ثانویه مانند تنوین، تشدید، مد و غیره وجود ندارد. همین تشتّت آراء در نگارش یا عدم نگارش برخی نشانه‌های ثانویه از سوی زبانمندان یک زبان، برای زبان‌آموزان نیز ایجاد ابهام می‌کند چرا که یک صورت واحد، استاندارد و مورد قبول در بین همه‌ی پژوهشگران و زبانشناسان حوزه‌های زبانی وجود ندارد. مخصوصاً در سطوح پیشرفته‌ی زبان‌آموزی گرایش به حذف این عناصر بالا می‌رود. با این وجود هرگونه انحراف از صورت صحیح و تنوین‌دار این دست کلمات را طبق شیوه‌نامه دستور خطّ مصوّب فرهنگستان، خطا محسوب کرده و به عنوان یک مقوله‌ی خطایی در پیکره نشان‌گذاری می‌گردد، البتّه به دلیل اهمیّت کمتر آن‌ها نسبت به خطاهای نشانه‌ی اصلی، آن‌ها را تحت عنوان خطاهای نشانه‌ی ثانوی جدا کرده‌ایم تا برای محقّقان و آزمونگران قابل تفکیک بوده و با سهولت بیشتری به بررسی خطاهای اصلی بپردازند. کد مربوط به این مقوله با توجه به معادل آن حرف (Tan[1]) است. در ادامه با توجّه به فرآیندهای خطایی گوناگون رخ داده بر روی این خطا، سایر کدها نیز معرّفی شده‌اند.

الف) خطای حذف تنوین

در این خطا زبان‌آموز از نگارش تنوین خودداری کرده و کلمه را به همان شکل ساده و فاقد نشانه‌ی تنوین می‌نویسد. بنابراین از آنجا که این نوع نگارش با دستور خطّ فارسی مغایرت داشته و در برخی موارد در تلفّظ کلمه ایجاد اشکال می‌نماید، خطا در نظر گرفته شده است. از این دست‌اند نگارش کلماتی مانند تقریبا، مثلا به جای صورت صحیح تقریباً و مثلاً. این خطا در پیکره توسط کد (Or,Dc-tan,M) نشان‌گذاری می‌شود.

[1] Tanwin

در بین خطاهای یافت شـــده در رابطه با تنوین به مواردی برخورد نموده‌ایم که در آن‌ها حذف تنوین نه تنها به لحاظ خوانش کلمه ایجاد ابهام می‌کند بلکه به لحاظ معنائی نیز مفهومی متفاوت از مفهوم مورد نظر را به خواننده القا می‌کند. به عنوان مثال عدم نگارش تنوین در کلمه‌ی عقلاً ممکن است در تلفّظ این کلمه برای خواننده ایجاد اشکال نموده و به صورت واژه‌ی عُقَلا خوانده شود. واضح است که حذف تنوین در اینگونه موارد هم‌ارزش با سایر موارد نبوده و از اهمیّت بیشتری برخوردار است چراکه عنصری ضروری حذف شده است. در اینجا جهت نمایش تحلیلی دقیق‌تر از کدهای ترکیبی و مشترک بین حوزه‌ها استفاده شده است. به این ترتیب خطای تنوین در اینگونه موارد توســط کد مشــترک (O=S) یعنی دو حوزه‌ی املائی و معنائی، در کنار نوع مقوله و فرآیند خطایی نمایش داده می‌شود.

ب) خطای جایگزینی تنوین

هنگامی که نشانه‌ی تنوین با حرف یا نشانه‌ای دیگر جایگزین شود خطای جایگزینی تنوین رخ می‌دهد. به عنوان مثال زمانی که زبان‌آموز به جای استفاده از علامت تنوین معادل آوایی نزدیک به آن یعنی حرف (ن) را استفاده می‌کند خطای جایگزینی تنوین رخ می‌دهد. برچسب مربوط به این خطا کد (Or,Dc-tan,R) است.

ج) خطای جابه‌جایی در نگارش تنوین

یکی از رایج‌ترین خطاهایی که در نگارش تنوین مشاهده شد، ترتیب ناصحیح نگارش آن بود. به این معنی که تنوین در نگارش کلمه به کار برده می‌شد امّا در مکانی غیر از جایگاه صحیح خود. در این نوع خطا اکثراً تنوین در جایگاه ماقبل کرسی (الف) قرار داشت. برچسب مربوط به خطای جابه جایی تنوین کد (Or,Dc-tan,O) است. مانند کلمه‌ی جمیعًا در آیه‌ی زیر:

در مواردی هم مشاهده شده است که علاوه‌بر رعایت نکردن ترتیب نگارش تنوین، زبان‌آموز کرسی یا پایه‌ی (الف) را نیز حذف نموده است. در اینگونه موارد دو خطای حذف حروف و ترتیب ناصحیح نگارش تنوین به صورت همزمان رخ داده است که به شیوه‌ی زیر برچسب گذاری می‌گردد:

۲-۲-۲-۲- خطای نشانه‌ی تشدید

تشدید تکرار مکرّر یک صامت یکسان است در کلمه که به عنوان نشانه‌ای ثانویه در بالای حروف الفبای فارسی به کار می‌رود. در مورد نگارش تشدید نیز همانند همزه اختلاف نظرهای زیادی وجود دارد. حتّی امروزه شیوه‌ی آموزشی غالب در مدارس به گونه‌ای است که اغلب تشدید حذف شده یا اجباری در نگارش آن دیده نمی‌شود. همچنین نگاه بسیاری از زبانشناسان و دستوریان به تشدید نیز به گونه‌ای است که ضرورتی در نگارش آن نمی‌بینند. امّا در پژوهش حاضر به دو دلیل نگارش تشدید را ضروری در نظر گرفته‌ایم. نخست آنکه در شیوه‌نامه‌ی رسم‌الخطّ فرهنگستان که تحقیق حاضر از آن تبعیّت می‌کند، نشانه‌ی تشدید در کلماتی که نیاز بوده ثبت شده است و دیگر آن‌که علی‌رغم اینکه در حال حاضر توجّهی به نگارش یا عدم نگارش

تشـدید نمی‌شـود امّا این نشـانه بخشـی از رسـم‌الخطّ زبان فارسـی را تشـکیل داده و به همین دلیل نیاز اسـت زبان‌آموز با شیوه‌ی صحیح نگارش آن به عنوان عنصری در زبان فارسی آشنایی کافی داشته باشد. بنابراین در صورت مشاهده‌ی هرگونه مغایرت با صورت استاندارد نگارش این نشانه، کلمه‌ی مورد نظر به عنوان ساختی خطامند قلمداد می‌شود. کد (Ta[1]) نماینده‌ی مقوله‌ی تشدید است.

۳-۲-۲-۲- خطای نشانه‌ی مد

(الف) اوّلین حرف از الفبای فارسی است که به صورت‌های مختلفی همراه با مصوّت‌های گوناگون نوشته می‌شود. یکی از این گونه‌ها، (الف) همراه با مد است که اصطلاحاً به آن الف ممدوده می‌گویند. الف ممدوده در رسم‌الخطّ فارسی در آغاز کلمه یا در آغاز هجا به کار برده می‌شود. در رابطه با نگارش یا عدم نگارش مد نیز مانند همزه، تنوین و تشـدید اختلاف نظرهایی وجود دارد. امّا نگارش مد زمانی اهمیّت خود را نشـان می‌دهد که حذف آن یا افزودن زائد آن باعث تغییر معنا در اثر ابهام در خوانش کلمه شـود؛ به عنوان مثال واژه‌ی آرم اگر بدون مد نگاشته شود ممکن است به صورت اِرَم خوانده شود که معنایی کاملاً متفاوت با واژه‌ی مورد نظر دارد. در اینگونه موارد نیز به کمک کدهای مربوط به حوزه‌های ترکیبی اهمیّت اینگونه خطاها را می‌توان نمایش داد. بنابراین از آنجا که در این نمونه یک عامل املائی منجر به رخ دادن یک خطای معنایی شده است بهتر آن اسـت که از کدهای ترکیبی دو حوزه‌ی املایی و معنایی یعنی کد (O=S) در سـطح اوّل تحلیل استفاده شود. کد در نظر گرفته شده برای زیرمقوله‌ی نشانه‌ی مد حرف (Ma[2]) است.

۳-۲-۲- خطای صورت یا فرم

مقوله‌ی سوم در خطاهای املایی، گروه خطاهای فرم هستند. به خطاهایی که منجر به تولید صورت‌های نابه هنجار در زبان فارسـی می‌شـوند به گونه‌ای که این صـورت‌ها در رسـم‌الخطّ زبان فارسـی وجود خارجی نداشته باشند خطای فرم اطلاق نموده‌ایم. به عنوان مثال نگارش دندانه‌های اضافی، افزایش یا کاهش نقطه به صـورت نامتعارف و غیره همگی خطاهای فرم هسـتند. از دیگر مواردی که در زمره‌ی این خطا قرار می‌گیرد واژگان و ترکیبات غیر فارسـی‌ای هسـتند که به ندرت رخ می‌دهند. این ترکیبات وارد زبان فارسـی شـده و با خود قواعد زبان مبداء را همراه دارند. از آنجا که بررسی چنین خطاهایی خارج از حوزه‌ی تحلیلی زبان فارسی است، صورت‌های مغایر با صورت اصلی و صحیح را خطای فرم در نظر گرفته و صرفاً از دید فرم و صورت به تحلیل این خطاها پرداخته می‌شـود. حرف (F) کد در نظر گرفته شـده برای مقوله‌ی خطای فرم اسـت و در ادامه زیرمقولات مربوط به آن شرح داده شده‌اند.

۱-۳-۲-۲- خطاهای مرزبندی

در رابطه با خطاهای مرزبندی اعم از سرهم نویسی یا جدانویسی کلمات در زبان فارسی عقاید گوناگونی

---

[1] Tashdid
[2] Mad

وجود دارد امّا همانطور که پیش از این گفته شـد آنچه که در این تحقیق مدّ نظر اسـت پیروی از شـیوه‌نامه‌ی رسم‌الخطّ فرهنگسـتان زبان و ادب فارسـی اسـت. بدین منظور با الگوگیری از این شـیوه‌نامه، کلمات مرکّبی که برخلاف دسـتورالعمل فوق نگاشـته شده باشـند دارای ساختی خطامند بوده و به عنوان نوعی خطا در نظرگرفته می‌شوند. براین اساس دو نوع خطای سرهم نویسی و خطای جدانویسی کلمات به عنوان دو زیرمقوله‌ی خطای مرزبندی در نظر گرفته شده و تنها فرآیند خطایی جایگزینی در رابطه با این دو زیر مقوله مشاهده شده است. زیرمقوله‌ی خطای مرزبندی به وسیله‌ی کد (B^1) نشان داده می‌شود.

نکته‌ی مهم دیگر در باب خطای مرزبندی کلمات، این اسـت که در میان داده‌های تحلیل شـده‌ی مربوط به خطاهای مرزبندی، سه نوع خطای عمده یافت شد؛ دسته‌ی اوّل خطاهایی بودند که برخلاف قواعد سرهم یا جدانویسـی نوشـته شـده بودند مانند کلمات خطامند سـیب‌ای، بسـتگان‌شـان، پنجتن و همجور به جای کلمات سـیبی، بسـتگانشـان، پنج‌تن و همه جور که همانطور که گفته شـد به وسـیله‌ی کدهایی مجزا تعریف شـدند. دسـته‌ی دوم کلماتی بودند که علاوه بر انحراف از قواعد مرزبندی، وجه تمایز معناداری با دسـته‌ی اوّل داشـتند و آن این بود که این دسته کلمات در واقع به اجزایی تجزیه شده بودند که یا با در نظر گرفتن ملاحظات ریشه شـناختی و تاریخی، به تکواژهای سـازنده‌ی خود تجزیه شـده‌اند یا زبان‌آموز با شـنیدن این کلمات و با توجّه به اسـتنباط‌های شـخصـی خود از کلمه، آن‌ها را به صـورت ترکیبی از کلمات تشـکیل‌دهنده‌ی واژه‌ی مورد نظر می‌نوشـت. در واقع این کلمات نیز مانند کلمات نوع اوّل کلماتی بسـیط نبوده‌اند و در هر یک دو یا چند تکواژ وجود دارد امّا صورت نوشـتاری تفکیک شـده‌ی آن‌ها به تکواژهای سـازنده برخلاف قواعد نوشـتاریشان اسـت. از این دسـتاند کلماتی چون چه‌گونه (چگونه)، پای‌تخت (پایتخت)، اوّل‌این (اوّلین)، نه‌شـد (نشـد)، نه‌گذاشـتی (نگذاشـتی) و غیره. نوع دیگری از خطای مرزبندی مربوط به صورت‌های بسـیطی می‌شـود که تنها از یک تکواژ آزاد قاموسـی سـاخته شـده‌اند و قابل تفکیک به اجزای دیگر نیسـتند. در حالی که زبان‌آموز به اشـتباه اقدام به جدانویسـی این کلمات نموده اسـت. در کلماتی خطامندی چون که‌تاب (کتاب) و دسـتور(دستور) این جداسازی رخ داده است.

سـرهم یا جدانویسـی خطامند برخی کلمات گاه منجر به تولید کلماتی می‌شـود که به لحاظ دسـتوری نیز مقوله‌ی کلمه را تغییر می‌دهند. به عنوان مثال اگر واژه‌ی (خصـوصـیات) به صـورت جدا نوشـته شـود یعنی معادل واژه‌ی (خصـوصـی‌ات)، در اینجا (ات) نه تنها دیگر نقش نشـانه‌ی جمع را از دسـت داده بلکه نقش جدیدی به عنوان ضمیر ملکی متّصل پذیرفته است. تغییر نحوی رخ داده در خطای املائی سرهم/ جدا نویسی را می‌توان توسط کد ترکیبی مربوط به دو حوزه‌ی املائی-نحوی یعنی کد (Or-Sx) نشان داد.

در برخی موارد نیز فاصـله‌ی کم یا زیاد موجب تولید خطای مرزبندی از نوع فاصـله می‌گردد. به عنوان مثال اگر فاصـله‌ی مابین اجزای یک کلمه بیش از فاصـله‌ی واقعی باشـد به طوری که بتوان آن‌ها را جدا از یکدیگر خواند، مانند جفت واژه‌های (ما در) یا (در هم) در ازای تولید کلمات مادر و درهم، آنگاه فاکتور

[1] Boundary

خطای معنایی نیز به خطای املائی رخ داده افزوده می‌شود و یا حتّی بالعکس اگر فاصله‌ی واقعی حذف شود و کلمات بیش از حدّ نیاز به یکدیگر بچسبند مانند تولید واژه‌ی وبا به جای دو کلمه (و با) در اینصورت نیز خطا به وسیله‌ی حوزه‌های ترکیبی املائی-معنائی یعنی کد (Or-Sx) برچسب‌دهی می‌شود.

۲-۳-۲-۲- خطای دندانه

دندانه‌ها یکی از پرکاربردترین مولّفه‌های نگارشی در رسم‌الخطّ زبان فارسی هستند به طوری که به سختی می‌توان کلمه‌ای را یافت که بدون دندانه نوشته شود. همین امر نیز موجب می‌شود تا گاهاً در نگارش دندانه‌ها خطا رخ دهد. خطای نگارش دندانه محدود به دو فرآیند افزایش و کاهش دندانه می‌شود که اگر کد (Da[1]) را معرّف این زیرمقوله بدانیم آنگاه دو برچسب به شرح زیر خواهیم داشت.

۳-۳-۲-۲- خطای نقطه گذاری

در بالا و پایین بسیاری از حروف زبان فارسی حداقل یک نقطه و حداکثر سه نقطه وجود دارد. بدین ترتیب در یک کلمه ممکن است چندین نقطه وجود داشته باشد. در این خطا نیز همانند خطای دندانه تنها دو فرآیند خطایی حذف و افزایش مشاهده شده است. خطای نقطه‌گذاری را با کد(Do[2]) به عنوان زیرمقوله‌ای برای خطای صورت در نظر گرفته‌ایم.

۴-۳-۲-۲- خطای نگارش ناصحیح الف و لام (ال)

الف و لام (ال) در زبان عربی پرکاربردترین حرف ربطی است که موجب می‌شود اسامی، معرفه یا شناس گردند. از آنجا که در زبان فارسی قرض‌گیری واژگانی از زبان عربی بسیار زیاد است در پی آن کلمات بسیاری وارد این زبان شده‌اند که دارای الف و لام بوده و به این ترتیب حرف تعریف (ال) وارد زبان فارسی شده است. همانطور که در تعریف در داده‌های بررسی شده به دو مورد خطای مربوط به مقوله‌ی (ال) برخورد نمودیم که شامل حذف و افزودن زائد آن است. این خطا را با در نظر گرفتن کد (Al[3]) در کنار مقوله‌ی اصلی و حوزه‌ی املایی، به صورت کد (Or,F-al) نمایش می‌دهیم.

۴-۲-۲- خطای علائم نگارشی

علائم نگارشی مجموعه‌ای از نشانه‌های قراردادی هستند که جهت انتقال برخی از مفاهیم و حالات از گونه‌ی گفتاری زبان به گونه‌ی نوشتاری به کار می‌روند. این نشانه‌ها به عنوان نشانه‌های مازاد بر حروف الفبای فارسی درگونه‌ی نوشتاری مورد استفاده قرار می‌گیرند. هر یک از این نشانه‌ها در زبان فارسی کاربرد یا کاربردهای خاصّ خود را دارند و به کار بردن صحیح هر یک از آن‌ها مستلزم آشنایی و اطّلاع از قواعد کاربردیشان است. لذا لازم است حین روند آموزش زبان در کنار دیگر مباحث زبان آموزش داده شوند نقطه، ویرگول، علامت تعجّب، گیومه، دونقطه، خط فاصله از جمله‌ی این علائم هستند. چناچه زبان‌آموز نتواند هر

---

[1] Dandaneh
[2] Dot
[3] Alif Lam

یک از این علائم را متناسب با کاردبرد خاصّ خود و در جایگاه مناسب به کاربرد، خطای املایی از نوع مقوله‌ی خطای علائم نگارشی رخ می‌دهد. این حوزه‌ی خطایی را نیز مانند سایر حوزه‌ها می‌توان به وسیله‌ی چهار فرآیند اصلی خطایی یعنی افزایش،کاهش، جابه‌جایی و جایگزینی تبیین نمود. کد مربوط به این مقوله، برگرفته از حروف اوّل معادل لاتین آن کد (Pu) در نظر گرفته شده است که با سایر کدهای مربوط به فرآیندهای خطایی ترکیب شده و برچسب‌های جدیدی را تولید می‌نمایند.

۳-۲- برنامه INCEPTION

INCEPTION یک محیط حاشیه نویسی متن است که برای انواع مختلف وظایف حاشیه نویسی روی متن نوشته شده مفید است. حاشیه نویسی معمولاً برای نگرانی های زبانی و/یا یادگیری ماشینی استفاده می شود. INCEPTION یک برنامه وب است که در آن چندین کاربر می‌توانند روی یک پروژه حاشیه نویسی کار کنند و می‌تواند همزمان چندین پروژه حاشیه نویسی را شامل شود. این یک سیستم توصیه‌کننده برای کمک به ایجاد حاشیه‌نویسی سریع‌تر و آسان‌تر فراهم می‌کند. علاوه بر حاشیه نویسی، می‌توانید با جستجو در یک مخزن اسناد خارجی و افزودن اسناد، یک مجموعه ایجاد کنید. علاوه بر این، می توانید از پایگاه های دانش استفاده کنید، به عنوان مثال برای کارهایی مانند پیوند موجودیت‌ها. این محیط توسط گروه زبان‌شناسی دانشگاه اصفهان معرفی و استفاده شد که برای داده‌های منحصر به این پروژه نیز از آنها استفاده کرده‌اند. به منظور آشنایی با نحوه بدست آوردن داده‌های حال حاضر بهتر است به عملکرده این برنامه پرداخته شود.

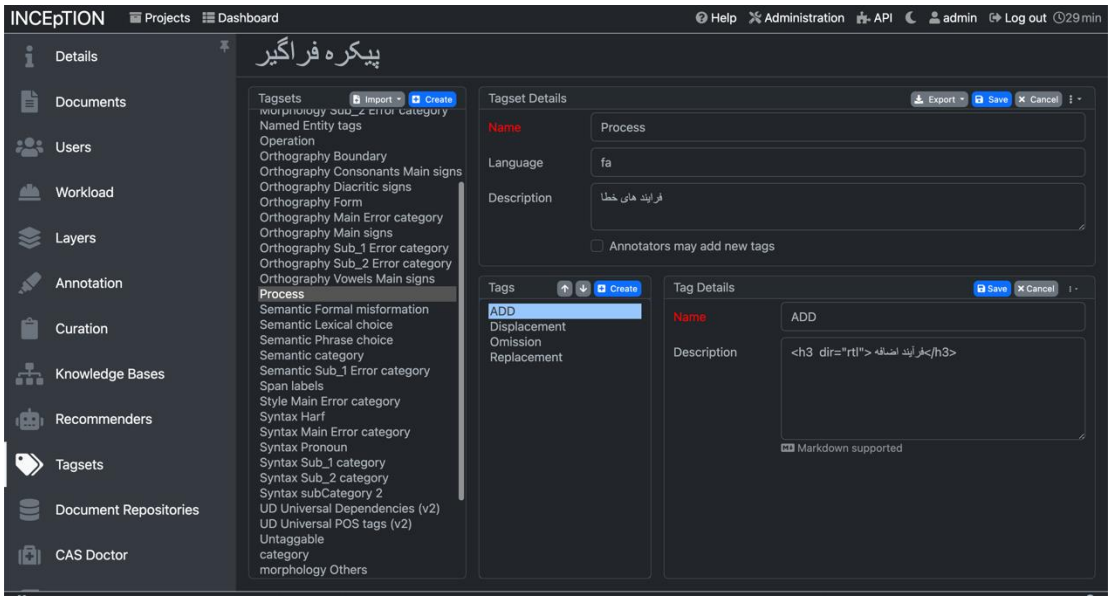
۱-۳-۲- نحوه عملکرد برنامه INCEPTION

برنامه INCEPTION [1] در دو محیط محلی و در بستر وب سایت به صورت آنلاین، رایگان در اختیار مخاطبین قرار گرفته است. اگر در محیط محلی برنامه را باز شود، می‌توان با نام کاربری admin و رمز عبور password وارد شد.

با ورود به برنامه می‌توان هم پروژه‌های قبلی را آپلود کرد و یا پروژه جدیدی را شروع کرد. در منوی سمت راست چهار انتخاب وجود دارد annotation , curation , monitoring و setting . در قسمت annotation در واقع همان عمل اصلی حاشیه گذاری و برچسب گذاری را انجام داد و در قسمت curation توسط کاربرهای دیگر بررسی مجدد روی قسمت های annotation انجام داد اما قبل از شروع به برچسب گذاری ابتدا باید لایه‌های تصحیح خطا را معرفی شود. برای تنظیم لایه ها در inception لازم است مشخص شود که نوع خطایی که این لایه پوشش می‌دهد، کدام

اســـت. اما از بین انواع زنجیره‌ای[۱]،فراداده[۲]، ارتباطی [۳]و طولی[۴]، ما نوع طولی را انتخاب می‌کنیم، زیرا کاربر میتواند یک رشته از واژه‌ها را انتخاب کند، دقیقا مانند هایلایت کردن بخشی از متن کار می‌کند.

سـپس باید مشـخص شـود که تا چه حدی می‌توان به جزئیات پرداخت، برای برچسـب گذاری در این لایه که خطاهای املائی می‌توانند در سـطح واج رخ دهند ما سـطح کاراکتر [۵]را انتخاب می‌کنیم. دیگر سـطح هایی که وجود دارند عبارتند از تک توکن[۶]، سـطح توکن[۷]، سـطح جمله[۸]. همچنین می‌توان اجازه داد که برچسـب‌های این لایه مرز بین جملات را نیز در نظر نگیرند. در نهایت باید برای این لایه برچســـب های خودش را تعریف کرد.



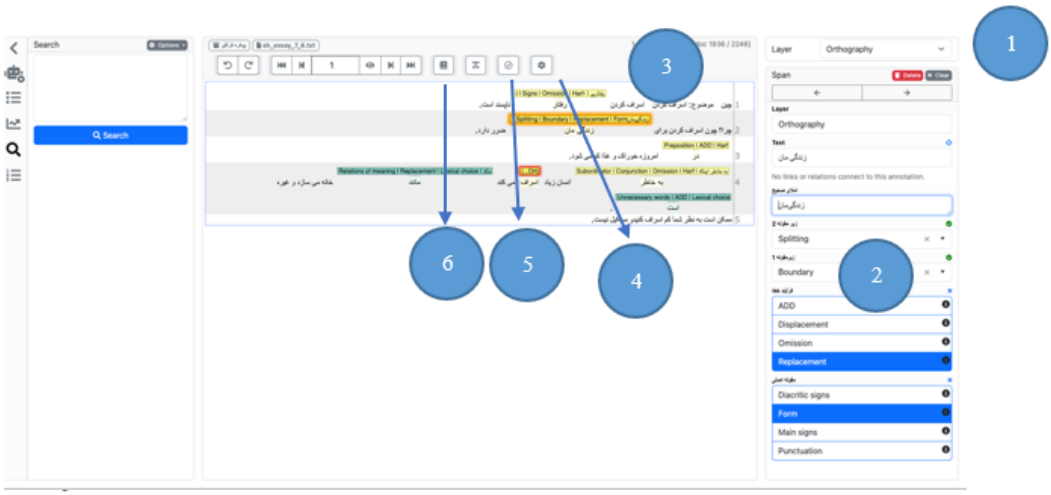
شکل ۲-۱ تنظیمات برچسب زنی در INCEPTION

برچسـب هایی که در نظر گرفته شـده بر اسـاس برچسـب های معرفی شـده توسـط گروه زبان‌شـناسـی اسـت با کمی تغییرات در نام‌گذاری برای درک راحت‌تر برچســـب‌ها و حذف چند برچســـب که امکان رخ دادن رای سـیسـتم های رایانه‌ای را نداشـتند مانند خطای کم یا زیاد بودن دندانه مجموعه برچسـب ها برای راحتی کار

[1] Chain
[2] Document MetaData
[3] Relation
[4] Span
[5] Character-level
[6] Single Token
[7] Token-level
[8] Sentence-level

برچسب‌گذاران براساس مقوله هایشان جدا شده‌است. اول از همه برچسب املای صحیح است که کاربر املای درست را در آن می‌نویسد و مقدار از پیش تعریف شده‌ای ندارد. سپس فرآیند خطاست که مقادیر مشخصی دارند (Add, Omission, Displacement, Replacement)، این برچسب ها که مقدار مشخص و از پیش تعیین شده ای دارند در قسمت Tagset پروژه تعریف می‌شوند.

برای لایه املایی علاوه بر فرآیند خطا سطح های دیگر تعریف شده که عبارتند از مقوله اصلی، زیرمقوله۱ و زیرمقوله۲ در اینجا مقوله اصلی ابتدا توسط کاربر انتخاب می‌شود و تا زیرمقوله۲ ممکن است فرآیند برچسب‌گذاری ادامه پیدا کند ولی انتخاب برچسب ها تا مرحله زیرمقوله ۱ و نوشتن عبارت درست خطا برای همه برچسب ها ضروری است،گرچه ممکن است خطایی باشد که زیرمقوله۱ نداشته باشد که در این صورت کاربر برچسب None را انتخاب می‌کند.



شکل ۲-۲ ترتیب مراحل برچسب زنی

۲-۳-۲- خروجی برنامه

خروجی این برنامه در فرمت فایل‌های xmi ساخته می‌شوند. فایل xmi (XML Metadata Interchanges) یک فرمت استاندارد مبتنی بر XML برای تبادل اطلاعات متادیتا بین ابزارها و سیستم‌های مختلف است. این فرمت عمدتاً برای توصیف مدل‌های UML[1] و دیگر متادیتاهای مربوط به توسعه نرم‌افزار استفاده می‌شود XMI .به توسعه‌دهندگان اجازه می‌دهد تا مدل‌های UML را بین ابزارهای مختلف مهندسی نرم‌افزار انتقال داده و از آن‌ها استفاده کنند.

خروجی‌های xmi حاوی چندین تگ میباشد که هرکدام یک اسم و چندین ویژگی دارند:

- Document Metadata : این تگ حاوی اطلاعات کلی در رابطه با فایل‌های ورودی است که شامل : id, language , title، آدرس محلی فایل، شماره تگ sofa که به این داده مربوط می‌شود.
- Sentence : این تگ‌ها نتیجه جداسازی جمله‌های فایل ورودی میباشد. هر تگ Sentence بیانگر هر جمله این فایل میباشد. این تگ حاوی صفت‌های: شماره sofa مربوطه، شروع جمله، پایان جمله و id خود تگ میباشد.
- Token : این تگ‌ها نتیجه جداسازی تمامی کلمه‌های فایل ورودی میباشد. هر تگ Token بیانگر هر کلمه این فایل میباشد. این تگ حاوی صفت‌های: شماره sofa مربوطه، شروع کلمه، پایان کلمه و id خود تگ میباشد.
- LayerDefinition : این تگ‌ها به تعداد لایه‌هایی که تولید کرده‌ایم ساخته میشوند و هرکدام بیانگر این است که هرلایه اسمش چیست و با توجه به اینکه خودمان تولیدشان کردیم کاربرد مخصوص به هرکدام را باید از قبل بدانیم.
- FeatureDefinition : این تگ‌ها ویژگی‌های هرلایه را توضیح میدهند و نامی که برای آنها انتخاب کرده‌ایم برای مثال زیرمقوله‌هایی که تعریف شده‌اند به عنوان ویژگی در هر لایه مربوطه اضافه میشوند.
- TagsetDataset : انواع برچسب‌های موجود که استفاده شده‌اند را نمایش میدهد. هرکدام در یک تگ مجزا.

[1] Unified Modeling Language

- Orthography : این تگ برای داده‌های ما کاملا حیاتی و حائز اهمیت اســت. این تگ برای هر غلط املائی استفاده میشود. ویژگی‌های این تگ شامل : شروع برچسب دهی، پایان برچسب دهی، مقدار درست واژه غلط، علت اتفاق این غلط که شامل Replacement,Add,Omission و displacement و ویژگی‌های main_category و sub_category که بیانگر نوع خطا هستند.
- Sofa : این تگ بیانگر کل ورودی برنامه است که حاشیه نویسی‌ها بر روی آن انجام شده‌است.
- View : این تگ نیز مانند sofa بیانگر کل متن ورودی است اما به صورت شماره گذاری شده که هر شماره id تگ توکن میباشد.

۴-۲- مدل‌های شبکه عصبی

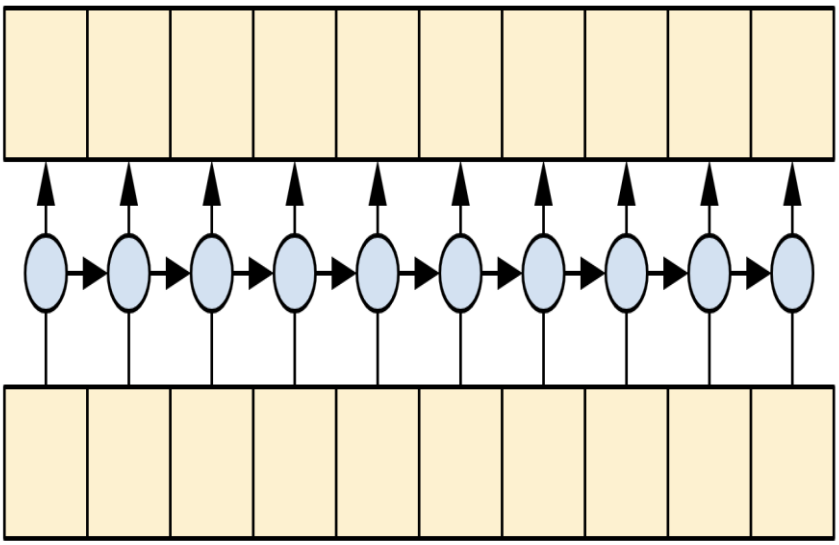
شبکه عصبی یا شبکه عصبی مصنوعی(Artificial Neural Network یاANN) نوعی مدل محاسباتی است که بر اساس نحوه عملکرد شبکه‌های عصبی زیستی مانند مغز انسان طراحی شده است. این شبکه‌ها از تعداد زیادی نود (که به آن‌ها نورون یا گره گفته می‌شود) تشکیل شده‌اند که به هم متصل شده و به‌صورت لایه‌بندی شده سازمان‌دهی می‌شوند.

شبکه‌های عصبی، به‌ویژه مدل‌های شبکه عصبی مصنوعی(ANN) ، ابزارهای قدرتمندی هستند که در حل مسائل پیچیده یادگیری ماشین مورد استفاده قرار می‌گیرند. یکی از کاربردهای پیشرفته و پرطرفدار این شبکه‌ها، مدل‌های دنباله به دنباله (Sequence-to-Sequence یا Seq2Seq) است. در این نوع مدل‌ها، هدف اصلی تبدیل یک دنباله ورودی به دنباله‌ای خروجی با طول متفاوت یا مشابه است.

مدل‌های Seq2Seq به‌طور گسترده در مسائلی مانند ترجمه ماشینی، تشخیص گفتار، تولید متن، و تصحیح املا به کار گرفته می‌شوند. در این مدل‌ها، یک شبکه عصبی ابتدا دنباله ورودی (مثل یک جمله) را پردازش کرده و سپس دنباله خروجی (مانند ترجمه یا نسخه اصلاح‌شده جمله) را تولید می‌کند.

این مدل‌ها معمولاً از معماری‌های پیچیده‌ای مانند رمزگذار-رمزگشا (Encoder-Decoder) استفاده می‌کنند که به کمک لایه‌های بازگشتی(RNN[1]) ، شبکه‌های عصبی کانولوشنی(CNN) ، و یا معماری‌های ترنسفورمر(Transformer) ، امکان یادگیری روابط زمانی و مکانی بین عناصر دنباله‌ها را فراهم می‌کنند. مدل‌های Seq2Seq به دلیل توانایی‌شان در پردازش و تولید دنباله‌های متنی یا عددی با ساختار پیچیده، به یکی از ابزارهای کلیدی در پردازش زبان طبیعی (NLP) و بسیاری از حوزه‌های دیگر تبدیل شده‌اند.

[1] Recurrent Neural Network



RNN -۲-۴-۱

شکل ۲-۳ معماری ساده RNN

شبکه عصبی بازگشتی[2] (RNN) نوعی از شبکه‌های عصبی است که برای پردازش داده‌های دنباله‌ای (Sequential Data)طراحی شده است. برخلاف شبکه‌های عصبی سنتی (مانند شبکه‌های عصبی پیش‌خور)، RNNها می‌توانند اطلاعات را در طول زمان حفظ کرده و وابستگی‌های زمانی را در داده‌ها مدل‌سازی کنند. این ویژگیRNN ها را به گزینه‌ای مناسب برای وظایفی که داده‌ها دارای ساختار زمانی یا ترتیبی هستند، تبدیل می‌کند.

ساختار و ویژگی‌های RNN :

- **حافظه داخلی :**تفاوت اصلی RNN با شبکه‌های عصبی معمولی این است که هر نود در RNN دارای یک حافظه داخلی است که وضعیت فعلی آن نود را در زمان‌های گذشته به یاد می‌آورد. به عبارت دیگر، خروجی هر نود در یک لحظه خاص به عنوان ورودی به نود در لحظه بعدی نیز داده می‌شود. این امر باعث می‌شود که شبکه بتواند اطلاعات قبلی را در زمان‌های بعدی استفاده کند.
- **بازگشت به خود :** در یکRNN ، هر نود در لایه بازگشتی به خودش در زمان بعدی متصل است. این

اتصالات بازگشتی به شبکه اجازه می‌دهد که اطلاعات زمانی را از ورودی‌های قبلی به یاد آورد و آن‌ها را در پردازش ورودی‌های جدید لحاظ کند.

کاربردهای RNN

- **پردازش زبان طبیعی(NLP)**

این شبکه ها در ترجمه ماشینی، تشخیص گفتار، تولید متن، و تحلیل احساسات بسیار مفید هستند.

- **پیش‌بینی سری‌های زمانی**

ازRNN ها می‌توان برای پیش‌بینی قیمت سهام، تقاضای انرژی، و سایر داده‌های سری زمانی استفاده کرد.

چالش‌های RNN :

- **مشکل گرادیان محو شونده یا انفجاری :**یکی از مشکلات اصلیRNN ها، مشکل گرادیان محو شونده یا انفجاری[۱] است که هنگام آموزش شبکه‌های بسیار عمیق یا دنباله‌های طولانی رخ می‌دهد. این مشکل باعث می‌شود که شبکه نتواند به درستی اطلاعات را از زمان‌های دور در حافظه خود نگه دارد. برای رفع این مشکل، معماری‌های بهبودیافته‌ای مانند LSTM[1] وGRU[2] معرفی شده‌اند که توانایی بهتری در یادگیری وابستگی‌های طولانی‌مدت دارند.

RNNها یکی از اجزای کلیدی در بسیاری از مدل‌های پیشرفته یادگیری ماشین هستند و نقش مهمی در توسعه سیستم‌های هوشمند و کار با داده‌های دنباله‌ای ایفا می‌کنند.

۲-۴-۲- Transformer

ترنسفورمرها (Transformers) یکی از پیشرفته‌ترین و تاثیرگذارترین مدل‌های شبکه عصبی در حوزه پردازش زبان طبیعی (NLP) و یادگیری عمیق هستند. این مدل‌ها، که برای اولین بار توسط پژوهشگران در مقاله‌ای به نام "Attention is All You Need"[3] در سال ۲۰۱۷ معرفی شدند، توانسته‌اند تحولی شگرف در عملکرد بسیاری از وظایف مرتبط با زبان و دنباله‌ها ایجاد کنند.

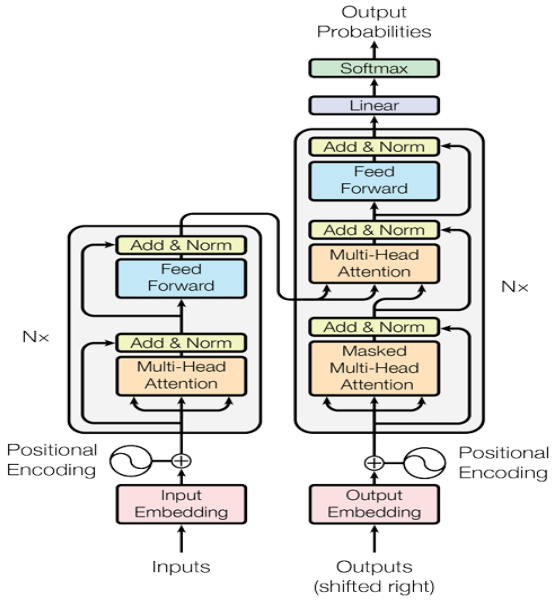
ترنسفورمرها بر پایه مکانیزم توجه (Attention Mechanism) کار می‌کنند که به مدل اجازه می‌دهد تا

[1] Long Short-Term Memory
[2] Gated recurrent unit

به تمام بخش‌های ورودی به‌طور همزمان نگاه کند و ارتباطات میان کلمات یا عناصـــر مختلف دنباله را بهتر درک کند. برخلاف مدل‌های قبلی مانندRNN ها و[4] LSTM ها که برای پردازش دنباله‌ها از یک ســـاختار زمانی و ترتیبی اســـتفاده می‌کردند، ترنســـفورمرها از موازی‌ســـازی پردازش‌ها اســـتفاده می‌کنند که منجر به افزایش سرعت و دقت در یادگیری می‌شود.

یکی از ویژگی‌های برجسته ترنسفورمرها، عدم نیاز به پردازش دنباله‌ها به ترتیب زمانی است که این امر باعث می‌شـود ترنســفورمرها بتوانند وابســتگی‌های طولانی‌مدت را بهتر مدیریت کنند و در کاربردهایی مانند ترجمه ماشینی، خلاصه‌سازی متن، تولید متن، و حتی تولید کد برنامه‌نویسی بسیار موفق عمل کنند.

۱-۲-۴-۲- معماری و ساختار ترنسفورمرها



شکل ۲-۴ معماری مدل transformer

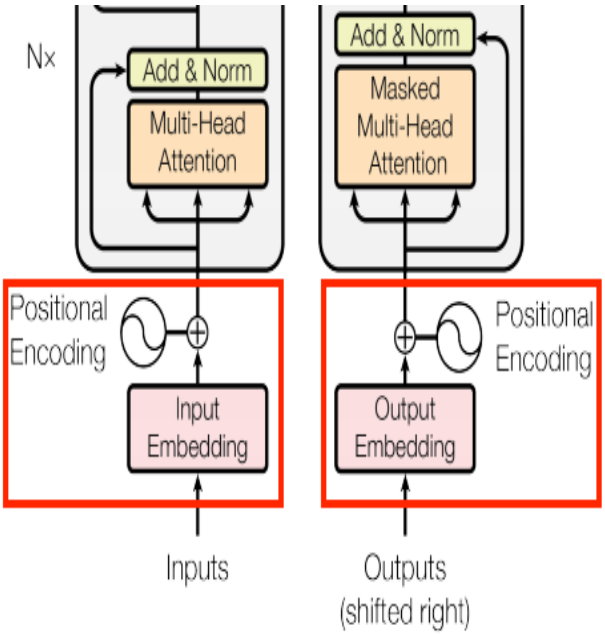
معماری ترنســـفورمر یکی از پیشـــرفته‌ترین و تأثیرگذارترین مدل‌ها در حوزه یادگیری عمیق، به‌ویژه پردازش زبان طبیعی(NLP[1]) ، اســـت. این معماری در ســـال ۲۰۱۷ در مقاله‌ای با عنوان "Attention is All You Need"توسط واسوآنی و همکارانش معرفی شد و توانست رویکردهای پیشین برای انجام وظایف دنباله به دنباله (مانند ترجمه ماشینی و خلاصه‌سازی متن) را به طور کامل متحول کند.

ترانسفورمر این معماری کلی را با استفاده از لایه‌های کاملاً متصل و کاملاً متصل برای رمزگذار و رمزگشا

[1] Natural language processing

که به ترتیب در نیمه‌های چپ و راست شکل ۲-۴ نشان داده شده‌اند، دنبال می‌کند.

- لایه کدگذاری و تعبیه:



شکل ۲-۵ لایه کدگذاری وتعبیه

ورودی‌های رمزگذار و رمزگشا از منطق کدگذاری جاسازی و موقعیتی یکسانی استفاده می‌کنند.لایه های توجه مورد استفاده در سراسر مدل ورودی خود را به عنوان مجموعه ای از بردارها بدون ترتیب می بینند. از آنجایی که مدل دارای هیچ لایه بازگشتی یا کانولوشنی نیست. به روشی برای شناسایی ترتیب کلمات نیاز دارد، در غیر این صورت دنباله ورودی را به عنوان نمونه ای از کلمات می بیند، چگونه هستید، چگونه هستید، چگونه هستید، و غیره قابل تشخیص نیستند. یک ترانسفورمر یک "رمزگذاری موقعیت" را به بردارهای تعبیه شده اضافه می کند. از مجموعه ای از سینوس ها و کسینوس ها در فرکانس های مختلف (در سراسر دنباله) استفاده می کند. طبق تعریف عناصر مجاور کدگذاری موقعیت مشابهی خواهند داشت.

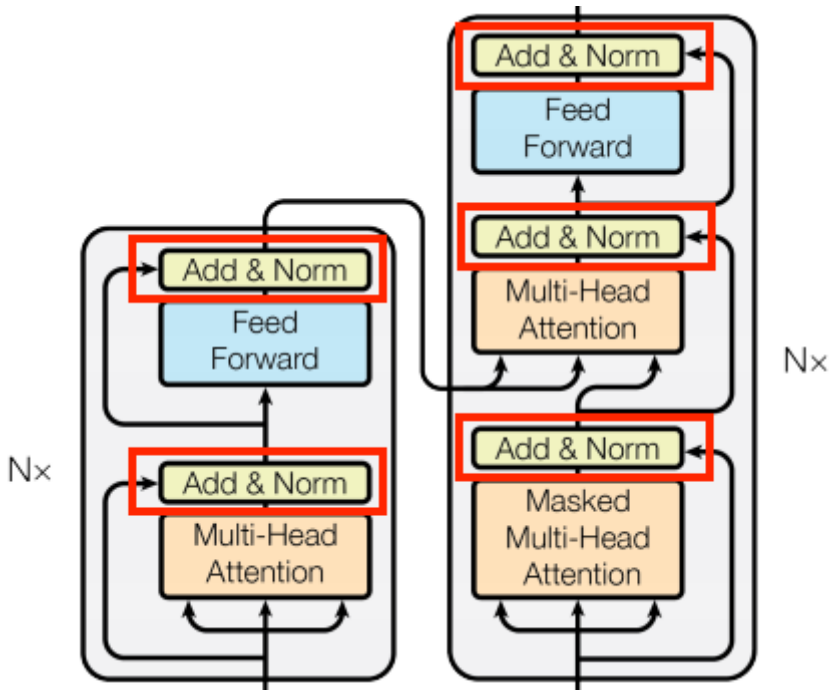
$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d_{model}})$$

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

تابع کدگذاری موقعیت، پشته ای از سینوس ها و کسینوس ها است که بسته به موقعیت آنها در امتداد عمق

بردار تعبیه شده، در فرکانس های مختلف ارتعاش می کنند. آنها در سراسر محور موقعیت ارتعاش می کنند.

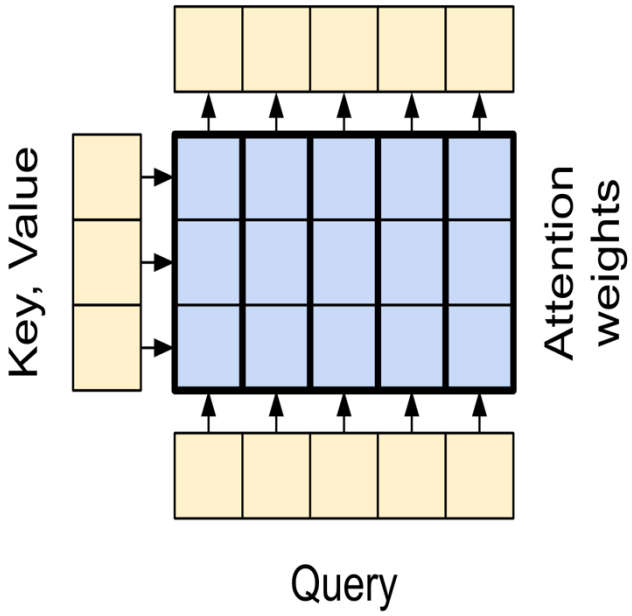
- Add and Norm: این بلوک‌های «افزودن و هنجار» در سراسر مدل پراکنده هستند. هر یک به یک اتصال باقیمانده می پیوندد و نتیجه را از طریق یک لایه LayerNormalization اجرا می کند. ساده



شکل ۲-۶ لایه Add and Norm

ترین راه برای سازماندهی کد در اطراف این بلوک های باقی مانده است. در بخش‌های زیر کلاس‌های لایه سفارشی برای هر کدام تعریف می‌شود. بلوک‌های باقی‌مانده «افزودن و هنجار» گنجانده شده‌اند تا آموزش کارآمد باشد. اتصال باقیمانده یک مسیر مستقیم برای گرادیان فراهم می کند (و تضمین می کند که بردارها به جای جایگزینی توسط لایه های توجه به روز می شوند)، در حالی که نرمال سازی مقیاس معقولی را برای خروجی ها حفظ می کند.

- The base attention layer: قلب معماری ترنسفورمر مکانیزم توجه است که به مدل اجازه می‌دهد تا ارتباطات میان کلمات مختلف یک جمله را بدون توجه به فاصله مکانی آن‌ها با یکدیگر، به‌طور مؤثر مدیریت کند. این مکانیزم باعث می‌شود که مدل بتواند بر روی بخش‌های مختلف ورودی تمرکز کند و وابستگی‌های طولانی‌مدت را به خوبی درک کند.



شکل ۲-۷ معماری attention

همانطور که در شکل ۲-۷ مشاهده می شود، در هر سطح از ترنسفورمر ها از ۳ لایه Attention استفاده شده‌است که پایه همه آنها از معماری Attention پیروی میکنند در عملکرد و کارکرد اما تفاوتی جزئی دارند. معماری لایه‌های Attention به صورت کلی در تصویر قابل مشاهده است.

در این لایه ها دو ورودی وجود دارد:

۱. دنباله پرس و جو : دنباله در حال پردازش ؛ دنباله ای که شرکت می کند (پایین).

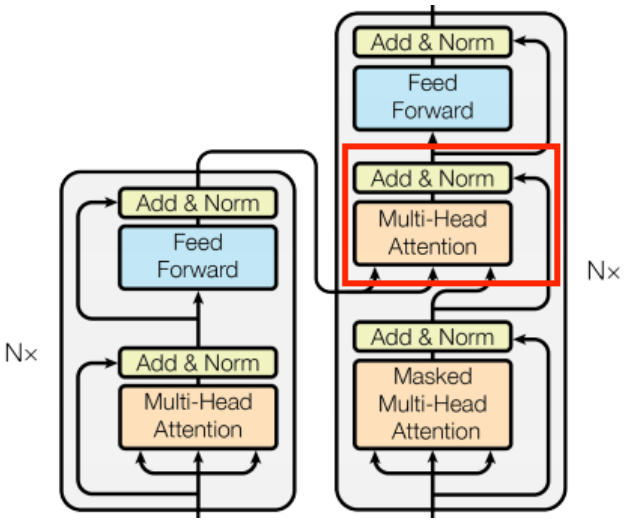
۲. دنباله متن : دنباله ای که در آن حضور دارد (سمت چپ).

خروجی به همان شکل دنباله پرس و جو است. مقایسه متداول این است که این عمل مانند جستجوی فرهنگ لغت است. Query نشان‌دهنده‌ی عنصر مورد نظر برای تمرکز است، Key نماینده ویژگی‌های تمام عناصر دیگر است، و Value اطلاعات واقعی‌ای است که در نهایت از آنها استفاده خواهد شد.

هنگامی که در یک فرهنگ لغت معمولی به جستجوی پرس و جو می پردازید ، فرهنگ لغت کلید تطبیق را پیدا می کند و مقدار مرتبط با آن را برمی گرداند .پرس و جو یا کلید تطبیق دارد یا اینطور نیست .شما می توانید یک فرهنگ لغت فازی را تصور کنید که کلیدها نیازی به مطابقت کامل ندارند. یک لایه توجه مانند این یک نگاه فازی را انجام می دهد ، اما فقط به دنبال بهترین کلید نیست .این مقادیر را بر اساس چگونگی مطابقت پرس و جو با هر کلید ترکیب می کند .این چگونه کار می کند؟ در یک لایه توجه ، پرس و جو ، کلید و مقدار هر بردار هستند .به جای انجام یک جستجوی هش ، لایه توجه ، پرس و جو و بردارهای کلیدی

را ترکیب می کند تا تعیین کند که چقدر با هم مطابقت دارند، .امتیاز توجه لایه میانگین را در تمام مقادیر باز می گرداند ، که توسط "نمرات توجه" وزن می شود. هر مکان، توالی پرس و جو یک بردار پرس و جو را فراهم می کند. توالی زمینه به عنوان فرهنگ لغت عمل می‌کند. در هر مکان در دنباله زمینه یک بردار کلید و ارزش را ارائه می‌دهد. از بردارهای ورودی به طور مستقیم استفاده نمی‌شود، لایه multiheadattionion شامل لایه هایی از Attention است که در واقع کلمات و عناصر ورودی را با کمک Attention هایی که در هر فضای معنایی بررسی می‌شوند، میتواند عناصر را در چند فضای معنایی متفاوت بررسی کند.

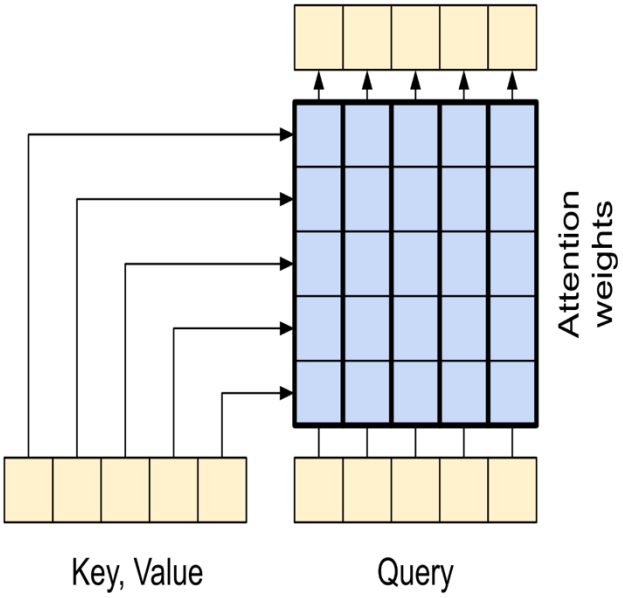
- The cross attention layer: تولنایی درک ارتباطات بین قطعات مختلف اطلاعات برای بسیاری از وظایف NLP بسیار مهم است .تصور کنید که یک نقد کتاب بنویسید - شما فقط متن را کلمه به کلمه خلاصه نمی کنید، بلکه بینش ها و ارتباطات بین فصل ها را ترسیم می کنید .توجه متقاطع را وارد کنید، مکانیزم قدرتمندی که پل‌هایی را بین دنباله‌ها ایجاد می‌کند و مدل‌ها را برای استفاده از اطلاعات از دو منبع مجزا توانمند می‌سازد.



شکل ۲-۸ لایه cross attention

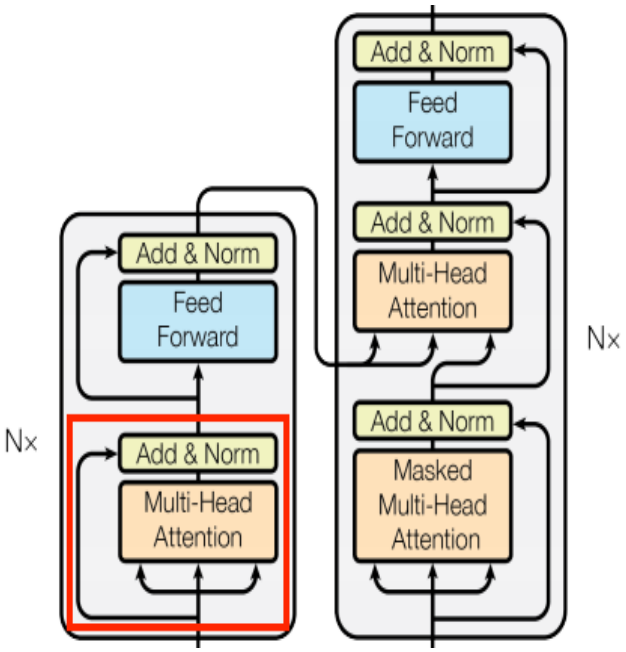
همانطور که در شکل – میبینید، این نوع توجه متقابل در لایه های رمزگشا استفاده میشود به این صورت که key و value ساخته شده از لایه رمزگزار را به عنوان ورودی خود میگیرد و با توجه به مفاهیمی که استخراج شده‌است تصمیم به ساخت خروجی متناظر با Query ورودی خود میکند.

مکانیسم او برای کارهایی مانند ترجمه ماشینی، خلاصه سازی و پاسخگویی به سؤال، که درک روابط بین توالی های ورودی و خروجی ضروری است، بسیار ارزشمند است.



شکل ۲-۹ معماری cross attention

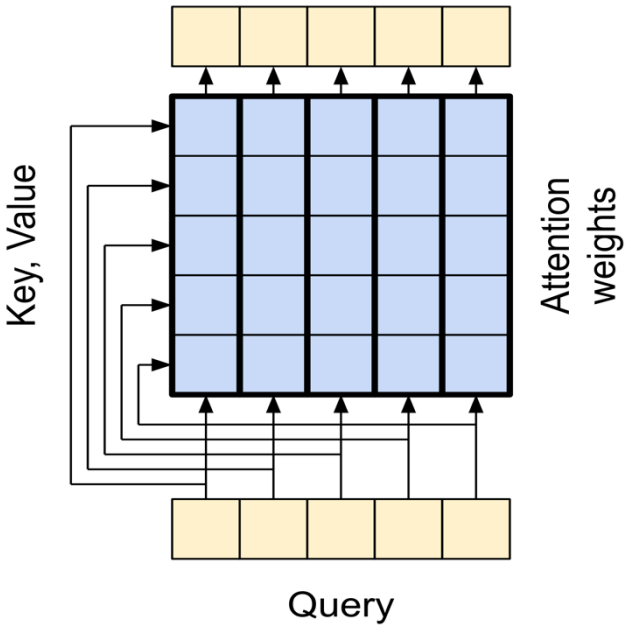
- Global Self-attention: مکانیسم‌های توجه با یک مبادله کلیدی روبرو هستند: گرفتن وابستگی‌های دوربرد در مقابل حفظ محاسبات کارآمد. این در دو رویکرد اصلی آشکار می شود: توجه جهانی و توجه محلی. تصور کنید که یک کتاب کامل را در مقابل تمرکز بر یک فصل خاص بخوانید. توجه جهانی کل دنباله را به یکباره پردازش می کند، در حالی که توجه محلی روی یک پنجره کوچکتر متمرکز می شود:
  - توجه جهانی: توجه جهانی وابستگی‌های دوربرد و زمینه کلی را جلب می‌کند، اما می‌تولند از نظر محاسباتی برای دنباله های طولانی گران باشد.
  - توجه محلی: توجه محلی کارآمدتر است اما ممکن است روابط دور را از دست‌بدهد.



شکل ۲-۱۰ لایه global attention

انتخاب بین توجه جهانی و محلی به عوامل مختلفی بستگی دارد:

- الزامات کار: کارهایی مانند ترجمه ماشینی نیاز به گرفتن روابط دور، جلب توجه جهانی دارند، در حالی که تجزیه و تحلیل احساسات ممکن است به نفع تمرکز توجه محلی باشد.
- طول توالی: توالی‌های طولانی‌تر توجه جهانی را از نظر محاسباتی گران‌تر می‌کنند و نیاز به رویکردهای محلی یا ترکیبی دارند.
- ظرفیت مدل: محدودیت‌های منابع ممکن است نیاز به توجه محلی حتی برای کارهایی که به زمینه جهانی نیاز دارند، داشته باشد.

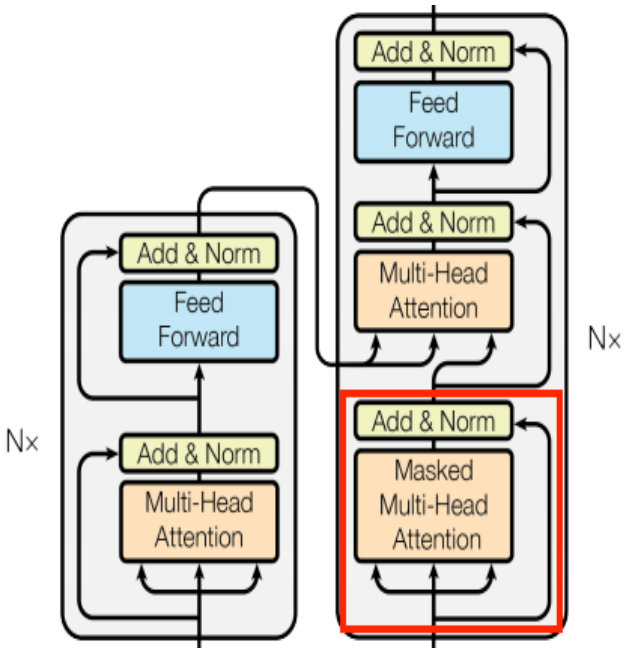


شکل ۲-۱۱ معماری global attention

برای دستیابی به تعادل بهینه، مدل ها می توانند از موارد زیر استفاده کنند:

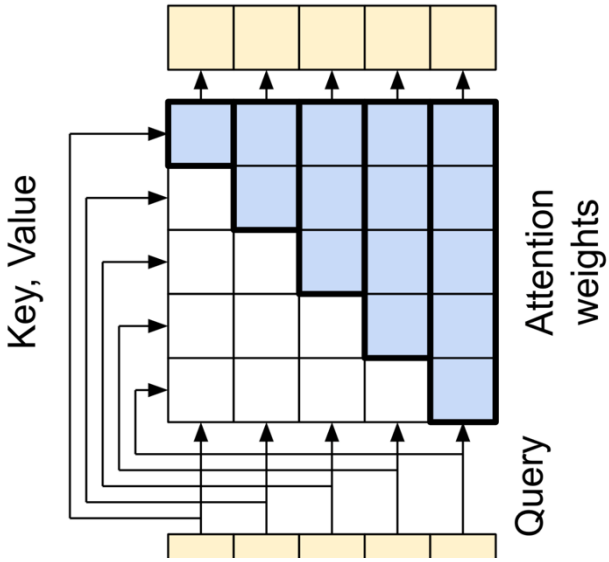
- سوئیچینگ پویا: از توجه جهانی برای عناصر کلیدی و توجه محلی برای دیگران استفاده کنید و بر اساس اهمیت و فاصله تطبیق دهید.
- رویکردهای ترکیبی: هر دو مکانیسم را در یک لایه ترکیب می کنند و از نقاط قوت مربوطه استفاده می کنند.

- The Causal Self-Attention: تصور کنید کلمه بعدی را در یک جمله بدون نگاه کردن به جلو پیش بینی کنید. مکانیسم‌های توجه سنتی با وظایفی که نیازمند حفظ نظم زمانی اطلاعات هستند، مانند تولید متن و پیش‌بینی سری‌های زمانی، مبارزه می‌کنند. آن‌ها به‌آسانی در سکانس «به جلو نگاه می‌کنند» که منجر به پیش‌بینی‌های نادرست می‌شود. توجه علّی با اطمینان از اینکه پیش‌بینی‌ها صرفاً به اطلاعات پردازش شده قبلی بستگی دارند، این محدودیت را برطرف می‌کند.



شکل ۲-۱۲ لایه causal self attention

یک ماسک خاص روی وزنه های توجه اعمال می شود و به طور موثر دسترسی مدل را به عناصر بعدی در دنباله مسدود می کند. به عنوان مثال، هنگام پیش‌بینی کلمه دوم در «پسر بچه‌ای که...»، مدل فقط می‌تواند «پسر» را در نظر بگیرد و نه «که » یا کلمات بعدی را.

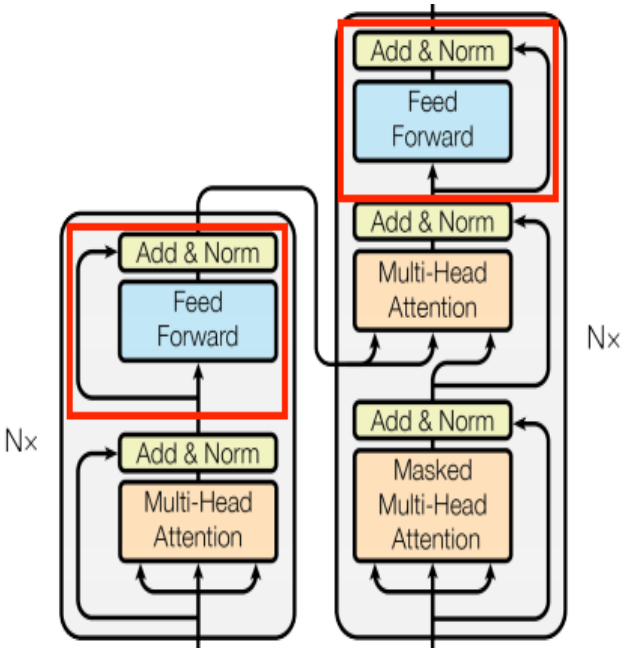


شکل۲-۱۳ معماری causal self attention

اطلاعات به صورت خطی جریان می یابد و نمایش هر عنصر صرفاً از عناصر ظاهر شده قبل از آن ساخته شده است. این مدل توالی را کلمه به کلمه پردازش می کند و پیش بینی هایی را بر اساس زمینه ایجاد شده تا آن نقطه ایجاد می کند.

توجه علّی برای کارهایی مانند تولید متن و پیش‌بینی سری‌های زمانی حیاتی است، جایی که حفظ نظم زمانی داده‌ها برای پیش‌بینی‌های دقیق حیاتی است.

- The Feed Forward Network: به زبان ساده، یک شبکه پیشخور یک ساختار چند لایه است که در آن اطلاعات در یک جهت، از ورودی به خروجی جریان می یابد. برخلاف شبکه‌های عصبی مکرر (RNN) که به حلقه‌های اطلاعاتی اجازه می‌دهند، شبکه‌های پیش‌خور داده‌ها را در یک خط مستقیم پردازش می‌کنند. در زمینه ترنسفورمرها، شبکه فید فوروارد یک زیر واحد در هر لایه رمزگذار و رمزگشا است. خروجی را از لایه توجه به خود می گیرد، که روابط بین بخش های مختلف دنباله ورودی را می گیرد و آن را بیشتر تغییر می دهد.



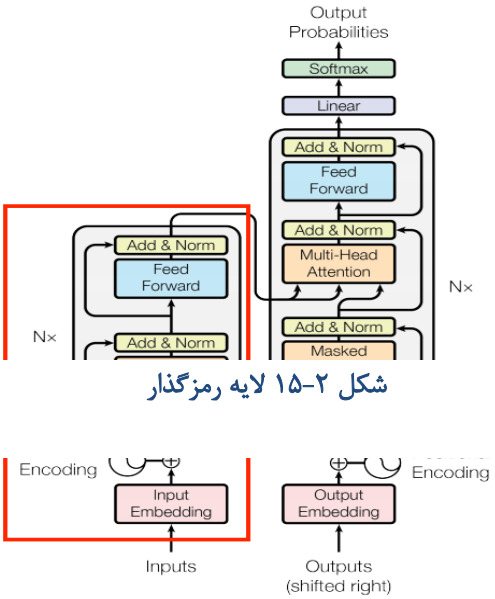
شکل ۲-۱۴ لایه‌های feed forward network

شبکه Feed Forward دو وظیفه اصلی را انجام میدهد:

دگرگونی غیرخطی: توجه به خود بازنمایی هایی با آگاهی از متن از ورودی ارائه میدهد. در ادامه این شـــبکه با اعمال توابع غیر خطی به این نمایش‌ها، لایه دیگر از پیچیدگی را اضـــافه می‌کند. این مدل اجازه میدهد تا الگوهای پیچیده تری را در داده‌ها بیاموزد.

ارتقاء ویژگی: این شـــبکه همچنین میتواند به عنوان راهی برای اصـــلاح ویژگی‌های اســـتخراج شـــده توســـط لایه توجه به خود دیده شـــود. می‌تواند بر ویژگی‌های مهم تاکید کند و موارد کمتر مرتبط را سرکوب کند و خروجی نهایی را آموزنده تر کند.

- Encoder: **Encoder**(رمزگذار) یکی از اجزای اصـــلی در معماری ترنســـفورمر اســـت که برای پردازش ورودی‌ها و اســـتخراج ویژگی‌های آنها طراحی شـــده اســـت. در معماری ترنســـفورمر، Encoderنقش مهمی در تولید نمایی از ورودی‌ها ایفا می‌کند که توســـط Decoder (رمزگشـــا) برای تولید خروجی‌ها اســـتفاده می‌شـــود. این معماری به‌ویژه در پردازش زبان طبیعی و ترجمه ماشـــینی بســـیار موفق عمل کرده است.



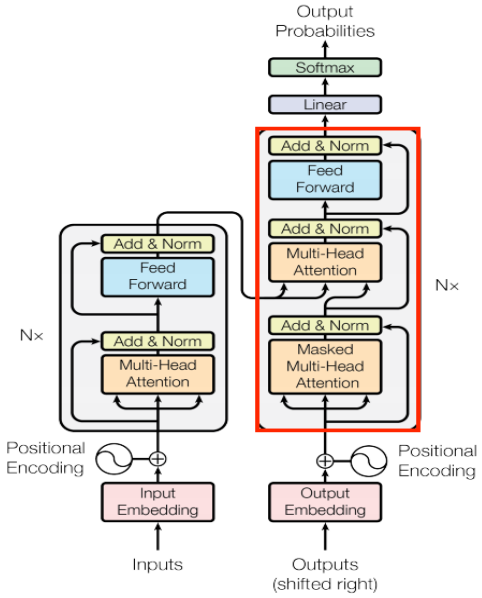
شکل ۲-۱۵ لایه رمزگذار

رمزگزارها در مدل های ترنسفورمری شامل چندین لایه رمزگزار می‌شوند که در نهایت به عنوان یک موجودیت جامع در مدل جا میگیرند.

- Encoder layer: هر لایه رمزگزار متشکل است از یک شبکه feed forward و یک attention ولی شامل لایه کدگزاری وتعبیه نیست. وجود چندین لایه رمزگزار به encoder کمک میکند که در فضاهای گوناگون شروع به جست و جو و کاوش کند.
- Decoder: **Decoder**(رمزگشا) یکی از اجزای حیاتی در معماری ترنسفورمر است که به‌طور خاص برای تولید دنباله‌های خروجی از یک ورودی رمزگذاری‌شده توسط Encoder طراحی شده است. این بخش از مدل ترنسفورمر به عنوان قلب فرآیند تولید در وظایف مختلفی مانند ترجمه ماشینی، خلاصه‌سازی متن، و پاسخ‌گویی به سوالات نقش‌آفرینی می‌کند.

همانند رمزگزار، رمزگشا در ترنسفرمرها شامل چندین لایه رمزگشا میشود که می‌تواند با احتساب جوانب متفاوت خروجی متناسب تر را تولید کند.

- Decoder Layer هر لایه رمزگشا شامل دو توجه و یک شبکه feed forward است بدون احتساب لایه کدگزاری. یعنی ورودی های تمامی این لایه‌ها یکسان است منتهی فضای جست و جو آنها متفاوت است.



شکل ۲-۱۶ لایه رمزگشا

۲-۵- مدل‌های نیمه آموزش دیده

بعد از انتشار مقاله " Attention is all you need" ترنسفورمرها شروع به پیشرفت کردند و انقلاب جدیدی در مدل‌های هوش مصنوعی seq2seq به وجود آمد. با پیشرفت مدل‌های seq2seq، مدل‌های جدید تر و با هایپرپارامترهای متفاوت تر و تعداد داده‌های بزرگتر آموزش دیدند و آماده برای استفاده هستند بعضی ها فقط از قسمت decoder ترنسفورمر استفاده میکنند مثل gpt و بعضی ها از قسمت encoder مثل bert و بعضی هردو. مدل‌های نیمه آموزش دیده مدل‌هایی بر این پایه هستند که با حجم عظیمی از داده آموزش دیده‌اند و هم آمادگی برای آموزش برای داده‌های خاص شما دارند و هم حاضر برای استفاده هستند. برای استفاده از این مدل‌ها در زبان فارسی میتوان به مدل‌های GPT2-FA[5] , XLM-R[6] , MT-5[7] , ParsBert[8] اشاره کرد. تفاوتی که این مدل‌ها با یکدیگر دارند این است که در عملیات متفاوت قادر به یادگیری هستند و چه بسا نوع Tokenizer ها نیز امکان دارد متفاوت باشد. برای مثال مدل ParsBert علی رغم XLM-R قادر به انجام عملیات Generative نیست و برای کارهایی مثل تحلیل احساسات و Token_Classification مناسب است. برای اطلاعات دقیق تر از مدل‌های نیمه آموزش دیده از لینک زیر میتوانید اطلاعات کسب کنید.

۶-۲- ارزیابی

مرحله‌ای که بعد از آموزش مدل‌های شبکه عصبی خیلی حائز اهمیت است، ارزیابی مدل است. معمولا ۲۰ درصد داده‌های موجود را به عنوان داده‌های تست جدا میکنند و بعد از آموزش مدل از داده تست جدا شده، تست می‌گیرند. معیارهای ارزیابی گوناگونی در حال حاضر در دنیای هوش مصنوعی تعریف شده‌اند و استفاده می‌شوند. انتخاب معیار مناسب می‌تواند ما را به ارزیابی و استنتاج بهتر هدایت کند. در این قسمت با پنج معیار ارزیابی که در پروژه استفاده شده‌اند آشنا می‌شویم.

۱-۶-۲- دقت

دقت (Accuracy) یکی از معیارهای رایج برای ارزیابی عملکرد یک مدل دسته‌بندی است. دقت نشان می‌دهد که چه نسبتی از نمونه‌ها به‌درستی توسط مدل پیش‌بینی شده‌اند (شامل پیش‌بینی‌های درست مثبت و پیش‌بینی‌های درست منفی).

رابطه محاسبه دقت:

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

- **TP[1]مثبت درست** : نمونه‌هایی که به‌درستی به عنوان کلاس مثبت پیش‌بینی شده‌اند.
- **TN[2]منفی درست:** تعداد نمونه‌هایی که به‌درستی به عنوان کلاس منفی پیش‌بینی شده‌اند.
- **FP[3]مثبت نادرست:** تعداد نمونه‌هایی که به اشتباه به عنوان کلاس مثبت پیش‌بینی شده‌اند در حالی که در واقعیت به کلاس منفی تعلق دارند.
- **FN[4]منفی نادرست** : تعداد نمونه‌هایی که به اشتباه به عنوان کلاس منفی پیش‌بینی شده‌اند (در حالی که در واقعیت به کلاس مثبت تعلق دارند.

۲-۶-۲- صحت

صحت (Precision) یکی از معیارهای ارزیابی مدل‌های دسته‌بندی، به‌ویژه در مسائل دسته‌بندی دودویی است. دقت نشان می‌دهد که از بین تمام نمونه‌هایی که به‌عنوان مثبت پیش‌بینی شده‌اند، چه نسبتی واقعاً مثبت هستند. به عبارت دیگر، دقت میزان صحت پیش‌بینی‌های مثبت مدل را اندازه‌گیری می‌کند.

---

[1] True Posetive
[2] True Negative
[3] False Posetive
[4] False Negative

رابطه محاسبه صحت :

$$Precision = \frac{TP}{TP + FP}$$

دقت به‌ویژه در مواقعی اهمیت دارد که هزینه مثبت کاذب (FP) بالاست. به عنوان مثال، در تشخیص بیماری، اگر مدلی تشـخیص دهد که فردی بیمار اسـت در حالی که او سـالم اسـت(FP) ، می‌تواند منجر به نگرانی و هزینه‌های اضافی شود. بنابراین، در چنین مواردی، مدلی با دقت بالاتر مطلوب‌تر است.

۳-۶-۲- بازیابی

بازیابی(Recall) ، که گاهی به آن **حساسیت (Sensitivity)** یا **نرخ تشخیص (True Positive Rate)** نیز گفته می‌شود، یکی از معیارهای مهم ارزیابی عملکرد مدل‌های دسته‌بندی است. بازیابی نشان می‌دهد که مدل چه نسـبتی از نمونه‌های مثبت واقعی را به‌درسـتی شـناسـایی کرده اسـت. به عبارت دیگر، بازیابی میزان توانایی مدل در یافتن تمامی نمونه‌های مثبت را اندازه‌گیری می‌کند.

رابطه محاسبه بازیابی:

$$Recall = \frac{TP}{TP + FN}$$

بازیابی به‌ویژه در مواقعی اهمیت دارد که از دست دادن یک نمونه مثبت (یعنی خطایFN) هزینه زیادی دارد. به عنوان مثال، در تشخیص سـرطان، شـناسـایی تمام بیماران مهم اسـت و از دسـت دادن حتی یک مورد می‌تواند عواقب جدی داشته باشد. بنابراین، در چنین مواردی، مدلی با بازیابی بالا مطلوب‌تر است.

۴-۶-۲- BLEU Score

BLEU score یکی از معیارهای معروف و پرکاربرد برای ارزیابی کیفیت ترجمه‌های ماشـینی یا مدل‌های تولید متن اسـت. این امتیاز در واقع برای مقایسـه خروجی‌های تولید شـده توسـط مدل با یک یا چند ترجمه مرجع (ترجمه‌های انسانی) استفاده می‌شود.

امتیاز BLEU یک معیار عددی است که نشـان می‌دهد چقدر یک ترجمه تولید شـده توسـط ماشـین به ترجمه مرجع (ترجمه انسـانی) نزدیک اسـت. این امتیاز عمدتاً بر اسـاس تطابق n-gram ها (توالی‌های n کلمه‌ای) بین ترجمه تولید شـده و ترجمه مرجع محاسـبه می‌شـود .n-gram ها می‌توانند تک‌کلمه‌ای (unigram)، دوکلمه‌ای(bigram) ، سه‌کلمه‌ای (trigram) و بیشتر باشند.

رابطه محاسبه **BLEU** :

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right).$$

که در آن:

- BP[1]همان جریمه طول است.
- Pn[2]دقت n-gram در سطح n است.
- wn[3]وزن n-gram هاست (معمولاً وزن‌ها برابر در نظر گرفته می‌شوند).

تفسیر امتیاز:BLEU

- امتیاز ۰ :به معنای هیچ‌گونه تطابق بین ترجمه تولید شده و مرجع نیست.
- امتیاز ۱ :به معنای تطابق کامل با ترجمه مرجع است (که عملاً در عمل به ندرت اتفاق می‌افتد).

تفسیر امتیاز BLEU :

- **امتیاز BLEU بین ۰ تا ۱۰ :**نشان‌دهنده عملکرد ضعیف مدل است. این امتیاز به این معناست که ترجمه‌های تولید شده توسط مدل اغلب با ترجمه‌های مرجع تطابق کمی دارند.
- **امتیاز BLEU بین ۱۰ تا ۳۰ :**نشان‌دهنده عملکرد متوسط است. مدل برخی از جنبه‌های جمله‌های مرجع را به درستی ترجمه می‌کند، اما هنوز دقت و روانی کافی را ندارد.
- **امتیاز BLEU بین ۳۰ تا ۵۰ :**نشان‌دهنده عملکرد خوب است. در این محدوده، ترجمه‌های تولید شده توسط مدل به طور کلی معنادار هستند و بسیاری از n-gram ها با ترجمه‌های مرجع تطابق دارند. برای بسیاری از کاربردهای عملی، این سطح از BLEU قابل قبول است.
- **امتیاز BLEU بالای ۵۰ :**نشان‌دهنده عملکرد بسیار خوب یا حتی عالی است. مدل توانسته است ترجمه‌هایی تولید کند که از نظر ساختاری و معنایی بسیار نزدیک به ترجمه‌های مرجع هستند. این امتیاز به ندرت در شرایط واقعی به دست می‌آید و اغلب نشانه‌ای از یک مدل بسیار قوی است.
- **امتیاز BLEU بالای ۷۰ :**به معنای کیفیت ترجمه‌ای است که تقریباً با ترجمه انسانی قابل مقایسه است. این سطح از BLEU اغلب به معنای تطابق بسیار بالای مدل با ترجمه‌های مرجع است، اما این اتفاق در عمل کمتر رخ می‌دهد.

[1] Brevity Penalty
[2] weight for n-gram
[3] n-gram modified precision

F1-Score -۲-۶-۵

F1-Scoreیک معیار ارزیابی است که به‌طور خاص برای اندازه‌گیری عملکرد مدل‌های دسته‌بندی، به‌ویژه در شرایطی که توزیع کلاس‌ها نامتوازن است، استفاده می‌شود F1-Score .ترکیبی از دو معیار مهم **دقت** (Precision)**و بازیابی** (Recall) است و به عنوان میانگین هارمونیک این دو محاسبه می‌شود. این معیار تعادلی بین دقت و بازیابی برقرار می‌کند و وقتی یکی از این دو معیار به تنهایی کافی نیست، F1-Score می‌تواند دید بهتری از عملکرد مدل ارائه دهد.

رابطه **f1-score**:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

F1-Score = 1 به این معناست که مدل دقت و بازیابی کاملی دارد؛ یعنی تمام پیش‌بینی‌های مثبت صحیح بوده و هیچ نمونه مثبت واقعی نادیده گرفته نشده است.

F1-Score = 0 به این معناست که یا دقت یا بازیابی مدل صفر است، که نشان می‌دهد مدل یا هیچ نمونه مثبتی را شناسایی نکرده یا تمام پیش‌بینی‌های مثبت آن اشتباه بوده‌اند.

۲-۷- معماری Web Application

معماری برنامه‌های تحت وب به مجموعه‌ای از اصول، الگوها، و ساختارهای طراحی اشاره دارد که برای توسعه و نگهداری برنامه‌های کاربردی مبتنی بر وب استفاده می‌شود. این معماری تعیین می‌کند که چگونه اجزای مختلف یک برنامه وب، از جمله سرور، پایگاه داده، رابط کاربری، و سایر بخش‌ها با یکدیگر تعامل دارند و به کاربران نهایی خدمات ارائه می‌دهند. هدف اصلی معماری وب این است که برنامه‌هایی مقیاس‌پذیر، قابل اعتماد، امن و آسان برای نگهداری و توسعه ایجاد کند. با توجه به اینکه برنامه‌های وب به‌طور فزاینده‌ای پیچیده‌تر می‌شوند، انتخاب معماری مناسب به توسعه‌دهندگان کمک می‌کند تا چالش‌های فنی و عملکردی را بهتر مدیریت کنند و تجربه کاربری بهتری ارائه دهند.

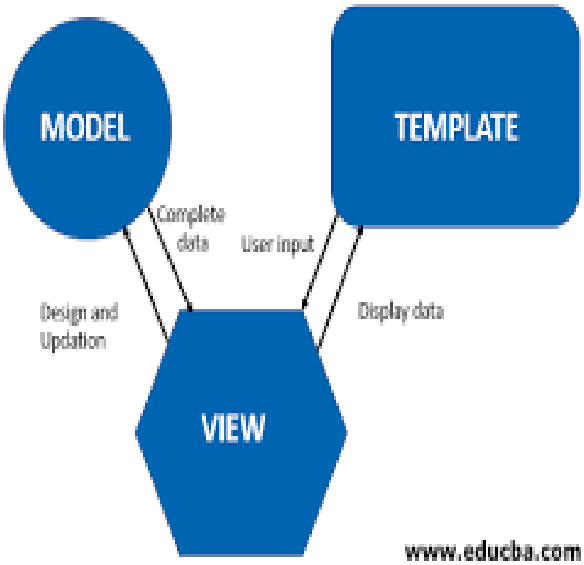
در اینجا به معماری MVT[1] که در پروژه استفاده شده‌است اشاره میکنیم.

MVT -۲-۷-۱

معماری MVT (Model-View-Template) یک الگوی طراحی است که توسط چارچوب جنگو (Django) در

[1] Model View Template

توسعه وب به‌کار می‌رود. این معماری شباهت زیادی به الگوی (Model-View-Controller) MVC[1] دارد، اما تفاوت‌هایی نیز دارد که آن را مخصوص به جنگو می‌کند.



شکل ۲-۱۷ معماری MVT

اجزای:MVT

۱. مدل Model :

مدل‌ها به عنوان لایه‌ای برای ارتباط با پایگاه داده عمل می‌کنند. هر مدل به یک جدول در پایگاه داده مربوط می‌شود و داده‌های برنامه را به صورت اشیاء (Object) در جنگو تعریف می‌کند. مدل‌ها شامل تعریف ساختار داده (مانند نوع داده‌ها و روابط بین آنها) و همچنین عملیات پایگاه داده (مانند درج، حذف، و بروزرسانی) هستند.

۲. **نما View :**

ویوها در جنگو منطق برنامه را کنترل می‌کنند و به درخواست‌های HTTP[2] پاسخ می‌دهند. وقتی کاربر یک درخواست به سرور ارسال می‌کند، ویو تصمیم می‌گیرد که چگونه به آن پاسخ دهد. این ممکن است شامل فراخوانی مدل‌ها برای دریافت یا تغییر داده‌ها و سپس بازگشت پاسخ مناسب باشد. ویوها همچنین ممکن است قالب‌های مناسب (Template) را برای ارائه اطلاعات به کاربر انتخاب کنند.

[1] Model-View-Controller
[2] Hypertext Transfer Protocol

۳. **قالب Template :**

قالب‌ها فایل‌های HTML[1] هستند که برای نمایش داده‌های به دست آمده از ویوها به کاربران استفاده می‌شوند. قالب‌ها در جنگو به شما امکان می‌دهند داده‌های دینامیک را با استفاده از زبان قالب‌سازی جنگو در صفحات وب قرار دهید. این به تفکیک منطق کسب‌وکار (در ویوها) از لایه نمایش کمک می‌کند، به‌طوری‌که توسعه‌دهندگان می‌توانند به راحتی طرح‌ها را تغییر دهند بدون اینکه به منطق برنامه دست بزنند.

۸-۲- جمع‌بندی

در این فصل با مفاهیمی از جمله تعریف و آشنایی با انواع خطاها که در توسط گروه زبان‌شناسی دانشگاه اصفهان معرفی شده‌اند آشنا شدیم و از این انواع غلط‌های املایی در برنامه INCEPTION برای حاشیه گذاری و برچسب گذاری توسط خانم دکتر متولیان استفاده شده‌است. سپس با داشتن و استخراج داده‌ها از فایل‌های خروجی INCEPTION، نیاز به معرفی و تهیه مدل شبکه عصبی داریم که در قسمت مدل‌های شبکه عصبی و مدل‌های نیمه آموزش دیده، به اختصار توضیح داده شده‌است. سپس به توصیف و ارزیابی و معیارهای ارزیابی پرداختیم و در نهایت یک معماری برای برنامه تحت وب را به اختصار بازگو شد.

[1] Hypertext Markup Language

فصل سوم
شرح پروژه

۳-۱- مقدمه

در این فصل به تفسیر و توضیح مراحل و اقداماتی که برای رسیدن به اهداف پروژه و کسب نتیجه انجام شده، پرداخته شده‌است. به دنبال تحقق اهداف پروژه مراحل متفاوتی پیموده شده که به تفصیل توضیح داده شده‌اند. این فصل با شناخت صورت مسئله و سپس انتخاب از میان راه‌حل‌های موجود و دلایل انتخاب هرکدام آغاز شده‌است. در پایان و در فصل چهارم با نتایج این راه‌حل‌ها ارائه شده‌است.

۳-۲- صورت مسئله

نخستین زبانی که هر فرد به آن تسلط کامل دارد، زبان مادری است. با گذر زمان ، رنگ و بوی این زبان در زندگی پررنگ‌تر شده و یادگیری زبان‌های جدیدتر را دشوارمیکند. آیا همه افراد در یادگیری زبان‌های جدیدتر با یک سری مشکلات و مسائل یکسان دست و پنجه نرم میکنند؟ زبان مادری چقدر در یادگیری زبان‌های خارجی موثر است؟ اشتباهاتی در هر فرد در یادگیری زبان جدید مرتکب می‌شود تا چه حد تحت تاثیر زبان مادری اوست؟ اگر پاسخ نخستین سوال بله در نظرگرفته‌شود، آیا مدل‌های هوش مصنوعی قادر به یادگیری این روابط و تقویت خود را دارند، یا خیر.

هدف از انجام این پروژه یافتن پاسخی برای پرسش‌های مطرح شده می‌باشد. این جنبه از سوالات در رابطه با زبان فارسی نیز بررسی شده و در حیطه غلط‌های املائی زبان فارسی آزمایش و خطا با مدل‌های هوش مصنوعی پرداخته شده.

۳-۳- داده

پس از استخراج داده‌های مورد نظر از فایل‌های XMI که پیش‌تر توسط گروه زبان‌شناسی دانشگاه اصفهان در اختیار این پروژه قرار گرفته بود، یک فایل csv منسجم ساخته‌شد که دارای ستون های زیر است:

۱. File_name: اسم فایلی که اطلاعات از آن استخراج شده‌اند.

۲. Begin : شروع کلمه نادرست.
۳. End : پایان کلمه نادرست.
۴. Wrong_word : کلمه نادرست
۵. Correct_wrod : کلمه تصحیح شده
۶. Wrong_sentence : جمله حاوی کلمه نادرست
۷. Correct_sentence : جمله تصحیح شده
۸. Process : عملی که باعث ایجاد خطا شده‌است.
۹. Main_category : زیرمقوله اصلی خطا
۱۰. Sub_category : زیرمقوله ۱
۱۱. Gender : جنیست زبان‌آموز
۱۲. Nationality : ملیت زبان‌آموز

این فایل داده شامل ۹۲۰۵ سطر می‌باشد که از این تعداد ۱۲۳ داده فاقد ملیت هستند (در فایل‌های اصلی ملیت ذکر نشده‌است). تعداد خطاهای در چهار دسته اصلی در جدول – طبقه‌بندی شده‌است.

جدول ۳-۱ آمار داده‌های ورودی و دسته‌بندی بر اساس نوع خطا

Punctuation	Main Signs	Form	Diacritic signs
2894	4839	1016	339

تعداد کل ملیت‌های موجود ۷۷ عدد است که داده‌ها با توزیع یکسان و نسبت مساوی پخش نشده‌اند به صورتی که برخی از ملیت‌ها مانند کویت و ازبکستان هرکدام با ۳ داده به عنوان داده‌های پرت و نامطئن شناسایی می‌شوند، در صورتی که هند با ۱۲۰۴ داده، بیشترین تعداد داده را به خود انتساب داده‌است.

آنالیز و نمودارهای تعداد غلط برای هر ملیت، استخراج شده‌است و بر روی یک CD ذکر شده‌است.

۳-۴- دسته بندی خوشه‌ای

قبل از اینکه وارد فازهای پیش‌پردازش و آموزش مدل شویم، سعی شده تا داده‌ها با استفاده از الگورتیم های متفاوت خوشه‌بندی، دسته‌بندی شوند. تا با بررسی نتایج ، امکان قرارگیری این داده‌ها در دسته‌های جداگانه مورد سنجش قرار گرفته‌شد. همچنین، به بررسی دقت مدل‌ها درتشخیص درست یا نادرست پرداخته‌شد. به تفسیر دیگر آیا این داده‌ها با ملیت متفاوت که به ابعادی که ناشی از الگوریتم T-NSE[1] کاهش یافته‌است، از یکدیگر جدا می‌شوند؟ یا خیر.

[1] t-distributed Stochastic Neighbor Embedding

$$z = \frac{x - \mu}{\sigma}$$

μ = Mean
σ = Standard Deviation

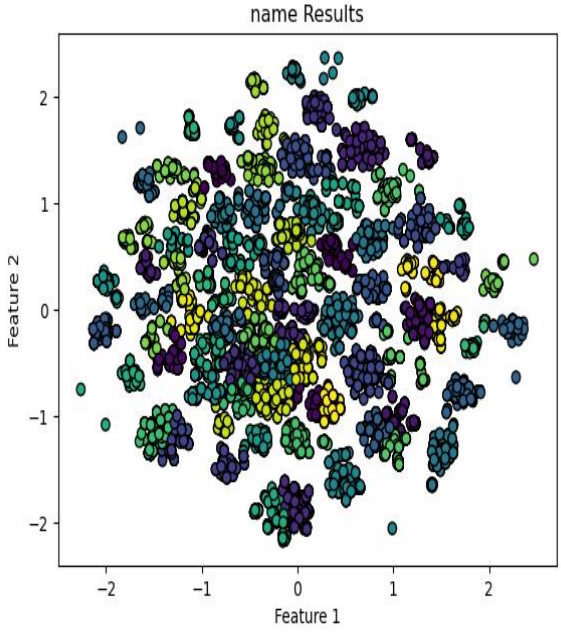
در این قسمت داده‌ها را برای نمایش گرافیکی بهتر و پردازش سریعتر توسط الگوریتم T-NSE به ۲ بعد کاهش دادیم. سپس داده‌ها را به صورت استاندارد، نرمالایز کردیم که توسط فرمول زیر محاسبه میشود:

سپس پس از پیش پردازش های انتخابی، داده‌ها را توسط الگوریتم هایی که در جدول – آمده‌است، محاسبه کردیم و ارزیابی را بر اساس معیارهای "Silhouette" , "Davies-Bouldin", "Calinski-Harabasz", "Adjusted Rank Index (ARI) ", "Normalized Mutual Information (NMI) " ارزیابی کردیم که در جدول زیر آمده‌است:

جدول ۳-۲ نتایج بدست‌آمده در دسته‌بندی بدون نظارت

	ARI	NMI	Sillouette	Davies-Bouldin	Calinski-Harabasz
KMeans	0.004	0.115	0.512	0.674	15179.191
Agglomerative	0.003	0.115	0.512	0.641	15004.180
DBSCAN	0.003	0.215	0.496	1.490	1005.742
Optics	-0.003	0.358	0.506	1.399	25.618
Brich	0.003	0.116	0.518	0.647	14895.262
MiniBatchKmeans	0.004	0.114	0.506	0.675	14677.073
Gaussian_Mixture	0.006	0.115	0.471	0.721	0.471

بهترین مدل بر اساس معیارهای جدول ۳-۲ آمده‌است.



شکل ۳-۱ نتیجه دسته‌بندی داده‌ها

همانطور که مشهود است، پیچیدگی داده‌ها زیاد است و همچنین تعداد برچسب‌های ملیت نیز برای این دسته از الگوریتم‌های بدون نظارتی نیز به کرات زیادتر. قطعا نمی‌شود از این دسته مدل‌ها نتیجه گرفت که به خوبی بتوانند دسته بندی درستی برای ملیت‌ها انجام دهند پس داده‌ها را با مدل‌های قوی‌تر می‌آزماییم. هرچند عملکرد و هدف مدل‌های آتی متفاوت است و قابل مقایسه با این سری الگوریتم‌ها نیستند.

۵-۳- پیش‌پردازش

برای آموزش مدل‌های هوش مصنوعی نیاز داریم که برای آموزش بهتر مدل، ابتدا چند عملیات پیش پردازش روی داده‌ها اعمال شود. ما برای پیش پردازش داده‌ها فقط از حذف نویز استفاده کردیم، حذف حروف نامربوطی که امکان داشت در جملات وجود داشته‌باشد مثل: '@', '#', '$', '%', '^', '&', '*', '(', و ... زیرا هدف ما از این مدل تصحیح خطا است و هرگونه تغییری مثل Stemming که کلمات را به حالت ساده تبدیل میکند و کلمه را تغییر میدهند، باعث خطا و تغییر نابه‌جای داده‌ها می‌شود. همچنین حذف علائم نگارشی نیز انجام ندادی؛ زیرا یک سری از داده‌های ما با برچسب غلط نگارشی معرفی شدند و قسمتی از تصحیح خطا ما وابسته به این علائم است. بعد از حذف این نویز ها نیاز داریم که جملاتی که در آنها کلمه غلط وجود دارد را به Tokenizer بدهیم و یک جمله را به صورت لیستی از کملات دریافت کنیم. ورودی را به صورت کلمه غلط و کلمه درست انتخاب نکردیم، زیرا به دنبال پیدا کردن معنی کلمات نیز هستیم که برای کلماتی که نوشتار آنها وابسته به متن است، دچار اشتباه نشود و با درک کردن معنی جمله و فضاهای جست و جو خود، کلمه درست

را انتخاب کند.

۶-۳- مدل Spell correction

برای استفاده و آموزش مدل‌های زبانی باید از مدل‌هایی استفاده کنیم که قابلیت آموزش دنباله به دنباله (seq2seq) داشته باشند. به این صورت که ورودی که جمله است را توسط یک tokenizer مناسب تبدیل به دنباله‌ای از کلمات کنیم، سپس توسط سیستم‌های مثل word_embedding[9] یا glove[10] به هر کلمه یک بردار با طول مناسب انتساب دهیم. روش‌های زیادی برای اینکار وجود دارد مثل TF-ITF[1] ، BOW[2]، one hot و ... . بعد از انتساب بردارهایی با طول یکسان به هر کلمه یک ماتریس (به ابعاد تعداد کلمات موجود در دیکشنری * بردار کلمه) بدست می‌آید که این ماتریس وزن‌های لایه اول شبکه قرار می‌گیرد همچنین میتوانیم با معرفی نوع عملکرد word_embedding این آموزش را به عهده خود شبکه عصبی بسپاریم. برای این کار نیاز داریم که یک دیشکنری نیز برای کل داده‌هایی که برای آموزش استفاده می‌شوند تهیه کنیم که به هر توکن موجود یک عدد انتساب میدهد (که همان اعداد بیانگر شماره سطر matrix_embedding متناظر با آن توکن است). همچنین در این دیکشنری نیاز است که توکن‌هایی نظیر , "<START>" , "<PAD[3]>" "<END>" قرار دهیم.

بعد از توکن بندی کردن جمله، نیاز داریم که طول تمام داده‌ها را به یک طول یکسان تبدیل کنیم. یعنی تمامی خروجی‌ها با یک طول یکسان و تمام ورودی‌ها نیز همچنین. برای اینکار طول بزرگترین دنباله را انتخاب میکنیم و هر دنباله‌ای که کمتر از آن بود، با توکن "<PAD>" پر میکنیم.

بعد از یکسان سازی دنباله‌ها با استفاده از دیکشنری به هر توکن عدد متناظر را انتساب جایگزین می‌کنیم و الان داده‌ها آماده آموزش هستند.

مدل‌هایی که در حال حاضر عملیات Spell correction را انجام می‌دهند، از ورودی‌های متفاوت‌تری استفاده می‌کنند. به این صورت که با داشتن کلمات درست با یک الگوریتم مخصوص روی داده‌ها نویز وارد می‌کنند و باعث ایجاد داده نادرست می‌شوند و با این کار هم می‌توانند حجم عظیمی از داده بدست بیاورند هم حالت‌های گوناگونی از اشتباه بودن یک کلمه را پشتیبانی می‌کنند.

در این پروژه چون داده‌ها توسط مصححان بدست آمده و همچنین ملیت نیز نقش مهم‌تری دارد باید داده‌های غلط توسط زبان‌آموزان تولید شوند، پس ما با حجم کمتری از داده سروکار داریم.

همچنین انتخاب یک توکن بندی مناسب که جملات را به کلمات، حروف، یا زیربخشی از کلمات تقسیم

---

[1] Term Frequency - Inverse Document Frequency
[2] Bag of Words
[3] Padding Token

کند بسیار نقش بسزایی در نتایج مدل دارد که در ادامه بررسی میکنیم.

ما برای آموزش، دانلود و تست مدل‌هایمان از بستر google colab[11] استفاده‌شد که با منابع ارزشمندی مثل 15G کارت گرافیک و 12G رم که در اختیار ما قرار میداد، کار آموزش را به شدت سرعت بخشید. مدل‌هایی که برای این پروژه در قسمت Spell correction استفاده شده‌اند به شرح زیر است:

۱-۶-۳- مدل نیمه آموزش دیده MT-5

مدل (Multilingual T5)MT5[1] یک نسخه چندزبانه از مدل -Text-to-Text Transfer) T5 Transformer) است که توسط محققان گوگل توسعه یافته است. این مدل برای کارهای مختلف پردازش زبان طبیعی (NLP) در زبان‌های متعدد طراحی شده است MT5 .بر اساس معماری ترانسفورمر Transformer) (است و مانند T5 تمام وظایف را به عنوان یک مسئله تبدیل متن به متن(text-to-text) در نظر می‌گیرد.

۱-۱-۶-۳- معماری MT-5

MT5 از معماری T5 استفاده می‌کند که شامل یک انکودر و یک دیکودر است. انکودر ورودی‌های متنی را به بردارهای ویژگی (feature vectors) تبدیل می‌کند و دیکودر این بردارها را به خروجی‌های متنی تبدیل می‌کند. درMT5 ، این معماری برای کار با زبان‌های مختلف بهینه‌سازی شده است و داده‌های آموزشی آن از مجموعه بزرگی از زبان‌ها تشکیل شده است.

ویژگی‌های کلیدی معماری:MT5

- **Text-to-Text:**تمامی وظایف (مثل ترجمه، خلاصه‌سازی، پاسخ به سوال، و غیره) به عنوان مسئله‌ای از تبدیل یک متن به متن دیگر دیده می‌شود.
- **چندزبانه :**مدل بر روی داده‌های چندزبانه آموزش دیده و قادر است به طور همزمان با زبان‌های مختلف کار کند.
- **پیش‌آموزش MT5 :**با استفاده از داده‌های چندزبانه، به صورت پیش‌آموزشی روی وظایف مختلفی مانند تکمیل جمله و ترجمه آموزش داده شده است.

۲-۱-۶-۳- Tokenizer MT-5

Tokenizerدر MT5 یک بخش حیاتی از مدل است که متن ورودی را به توکن‌ها (واحدهای کوچک‌تری از متن) تبدیل می‌کند. این توکن‌ها می‌توانند کلمات کامل، بخش‌هایی از کلمات، یا حتی کاراکترها باشند . Tokenizerبه کار رفته در MT5 بر اساس روش SentencePiece است که به ویژه برای کار با زبان‌های

[1] Multilingual T5

مختلف مناسب است.

ویژگی‌های کلیدی:Tokenizer MT5

- **SentencePiece:**از یک روش بدون نیاز به واژه‌نامه (vocabulary) ثابت استفاده می‌کند که به جای قطعه‌بندی دستی، با استفاده از مدل‌های آماری به تقسیم متن می‌پردازد.
- **زبان‌های مختلف** Tokenizer **:**به گونه‌ای طراحی شده است که با بسیاری از زبان‌های مختلف به طور همزمان کار کند و به همین دلیل توکن‌هایی تولید می‌کند که به خوبی برای زبان‌های مختلف مناسب است.

۳-۱-۶-۳- استفاده از مدل MT-5

همانطور که گفته‌شد، مدل MT-5 قابلیت پشتیبانی زبان‌هایی غیر از زبان‌های اینگلیسی دارد(۱۰۱ زبان) و ما ابتدا از این مدل نیمه‌آموزش دیده برای عملیات Spell correction پروژه استفاده کردیم. اما مشکلی که باعث میشد مدل به خوبی نتواند با این تعداد داده کم درصد خوبی دقت بدهد، این بود که زمانی که از این مدل استفاده میکردیم متعاقبا نیاز است که از Tokenizer مخصوص این مدل نیز استفاده کنیم. Tokenizer MT-5 کلمات و جمله‌هارا به زیر کلمه تبدیل و توکن بندی میکند. زمانی که به این صورت توکن بندی میکند، باعث می‌شود کلمه‌ای که نادرست است و به دو توکن زیرکلمه‌ای درست تبدیل کند و که ممکن در مابقی جمله پیدا شده‌باشد اما در جایگاه درست. اینچنین یک کلمه نادرست به دو زیر کلمه تبدیل می‌شود که امکان دارد در جاهای دیگر جمله درست باشد و این امر موجب میشد که مدل با این میزان داده دقت مناسبی ارائه ندهد. شاید اگر تعداد داده بیشتر بود امکان یادگیری عمیق تر و درک صحیح‌تر و مجزا سازی آنها برایش مهیا می‌شد.

۳-۶-۲- مدل نیمه آموزش دیده XLM-R[1]

مدل XLM-R یک نسخه چندزبانه از مدل RoBERTa است که توسط فیسبوک (Meta) توسعه یافته است. این مدل برای انجام وظایف مختلف پردازش زبان طبیعی (NLP) در زبان‌های متعدد طراحی شده و با استفاده از داده‌های بزرگ و متنوع در زبان‌های مختلف آموزش دیده است .به عنوان یکی از مدل‌های برجسته در پردازش زبان‌های چندگانه مطرح است و در بسیاری از وظایف، عملکرد بسیار خوبی ارائه می‌دهد.

[1] XLM-RoBERTa

۱-۲-۶-۳- معماری XLM-R

بر پایه معماری RoBERTa[1] است که خود یک نسخه بهینه‌سازی‌شده از مدل BERT می‌باشد. مانند BERT، XLM-Rاز معماری ترانسفورمر استفاده می‌کند و برای مدل‌سازی ارتباطات بین کلمات در یک متن طراحی شده است. مدل XLM-R دارای همان ساختار ترانسفورمر است، با لایه‌های متعددی از انکودر که وظیفه تبدیل ورودی‌های متنی به بردارهای ویژگی را دارند.

۲-۲-۶-۳- XLM-R Tokenizer

Tokenizerدر XLM-R از روش BPE[2] استفاده می‌کند، که یک روش محبوب برای قطعه‌بندی متن به توکن‌ها است. این روش با استفاده از ترکیب پر تکرارترین جفت‌های کاراکتر، توکن‌های نهایی را تولید می‌کند. این Tokenizer برای پشتیبانی از زبان‌های مختلف طراحی شده و به خوبی با زبان‌هایی که از الفبای غیرلاتین استفاده می‌کنند، سازگار است.

۳-۲-۶-۳- استفاده از XLM-R

ما از این مدل نیز استفاده کردیم اما همانند MT-5 متوجه شدیم که نتایج نادرستی که از این مدل بدست می‌آید، ناشی از توکن بندی نادرست است.

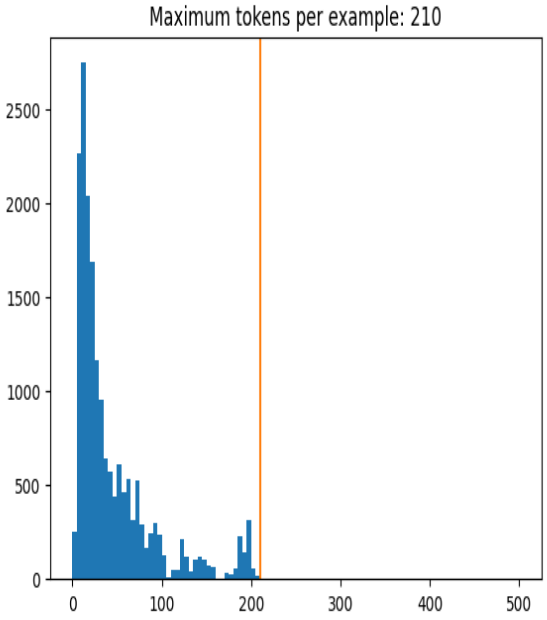
۳-۶-۳- مدل Transformer

در نهایت شروع به ساختن یک مدل Transformer کردیم. ابتدا کل داده‌ها را به نسبت ۸ به ۲ به صورت رندوم جداسازی کردیم و به عنوان داده Test و Train نامگذاری کردیم. سپس داده‌های Train که حاوی جملات غلط و جملات درست فایل استخراج شده‌بود، را با کمک Word_Tokenizer کتابخوانه Hazm که برای پردازش زبان فارسی ساخته شده‌است، توکن بندی کردیم. این تابع کتابخوانه جملات را به توکن‌هایی از جنس کلمه تبدیل میکند و در خروجی یک لیست برمی‌گرداند. از آنجایی که برای این مدل لایه positional embedding در نظر گرفتیم، دیگر نیاز به ساختن ماتریس embedding نیست و به عهده خود مدل میباشد پس برای ساختن دیتاست ورودی کافیست ابتدا یک دیکشنری از تمام کلمات دیتاست train بسازیم و بعد از یکسان سازی طول تمام داده‌ها با کمک توکن <PAD>، طول تمام داده‌ها را یکسان می‌کنیم. سپس با کمک دیکشنری که تولید کردیم داده‌ها را که لیستی از کلمات هستند را به لیستی از اعداد تبدیل میکنیم. با احتساب توکن‌های "<UNK[5]>" , "<eos[4]>" , "<sos[3]>" , "<PAD>" اندازه دیکشنری به ۹۶۸۷

[1] Robustly optimized BERT approach
[2] Byte-Pair Encoding
[3] Start of Sentence
[4] End of Sentence
[5] Unkwon Token

کلمه رسید.

برای انجام عملیات یکسان‌سازی طول داده‌ها یک آمار از طول تمام داده‌های Train گرفته و بیشترین اندازه را محاسبه شده‌است که در تصویر فوق نمایش داده شده‌است.

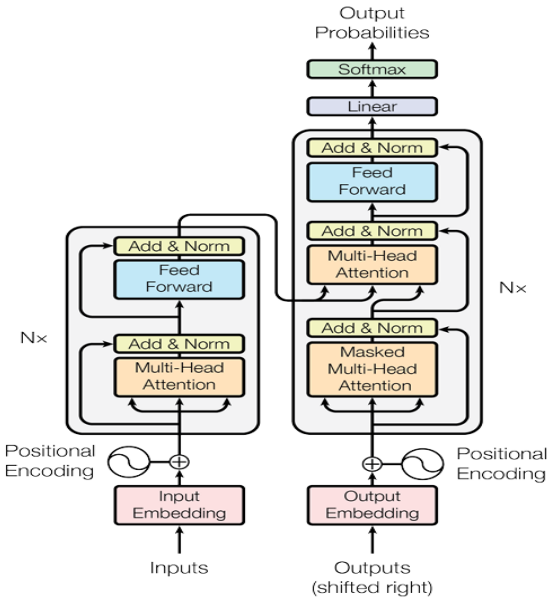


شکل ۳-۲ طول و تعداد داده‌های ورودی

بیشترین طول دنباله ورودی ۲۰۹ است. بنابراین طول تمام دنباله‌ها را با کمک Padding به ۲۵۶ می‌رسانیم.

۱-۳-۶-۳- لایه‌های مدل

این مدل با استفاده از کتابخانه‌های Tensorflow پیاده‌سازی شده‌است و کمک شایانی در ساخت مدل کرده‌است. ساختار این مدل شامل تمام لایه‌هایی که در مدل ترنسفورمر در شکل ۳-۴ قابل مشاهده است، می‌باشد.



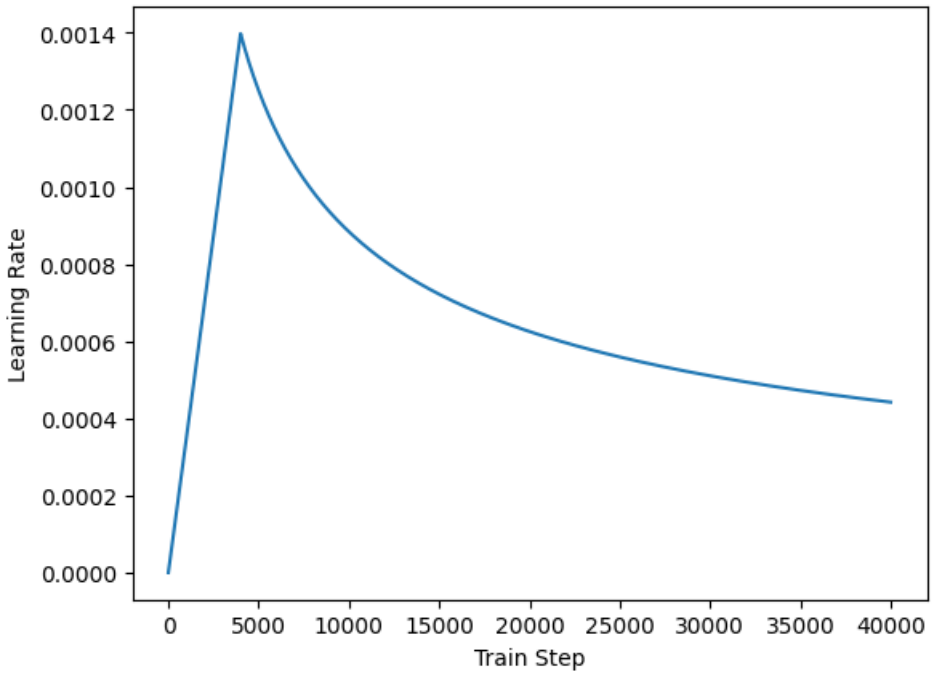
شکل ۳-۳ معماری مدل استفاده شده برای Spell correction

۲-۳-۶-۳- هایپرپارامترهای مدل

این مدل بر خلاف مدل ترنسفورمری که در مقاله "Attention Is All You Need" که ۶ لایه دارد، به دلیل کبود حافظه کارت گرافیکی از ۴ لایه استفاده میکند. همچنین این مدل همانند مدل ذکر شده در مقاله از ۸ Head استفاده می‌کند برای انجام عملیات‌ها به صورت موازی و بررسی معنا کلمات در فضاهای متفاوت معنایی. بعد ورودی مدل D_Model برخلاف مدل اصلی که ۵۱۲ می‌باشد، ۲۵۶ است زیرا برای پوشش داده‌های ما کافی است. بعد شبکه عصبی پیشرو (Feedforward) که با عنوان Dff[1] مطرح شده‌است را برای سادگی محاسبات ۵۱۲ قرار دادیم که در مدل اصلی ۲۰۴۸ است. نرخ Dropout نیز همانند مقاله ذکر شده، ۰/۱ در نظر گرفتیم.

بهینه‌سازی (Optimizer) که برای این مدل در نظر گرفتیم، همانند خیلی از مدل‌های هوش مصنوعی مطرح، Adam در نظر گرفتیم بخاطر عملکرد بهتری که ارائه می‌دهد. نرخ یادگیری نیز به صورت داینامیک تعریف می‌کنیم که با شروع پروسه آموزش با شیب زیاد، افزایش می‌یابد و سپس به صورت منحنی کاهش می‌یابد که در شکل – قابل مشاهده‌است.

[1] Dimension Feed Forward



شکل۳-۴ نمودار رشت نرخ یادگیری

تابع هدر رفت (Loss Function) را Sparse categorical cross entropy در نظر گرفتم که بدون احتساب توکن‌های "<Pad>" محاسبه می‌شود.

داده‌های Train را به دو قسمت Train و Validation با نرخ ۰/۲ تقسیم کردیم و در دسته‌های ۶۴ تایی در ۳۰ دوره (Epoch) به عنوان ورودی به مدل دادیم.

۳-۳-۶-۳- مدل آموزش‌دیده همراه با برچسب ملیت

برای آموزش مدل با احتساب ملیت و درک بهتر از رابطه ملیت و اشتباهات املائی مجبور به حذف غلط‌های Punctuation شدیم، زیرا علاوه بر اینکه علائم نگارشی در اکثریت زبان‌ها یکسان است، عمده این اشتباهات می‌تواند سهل انگاری نویسنده باشد. همچنین دسته غلط‌های Diacritic signs را نیز حذف کردیم به دلیل تعداد کم که ممکن بود باعث یادگیری ناقص مدل شود. پس در نتیجه بر روی دو دسته Main_Signs و Form تمرکز کردیم. در نهایت تعداد داده تست ما به ۴۷۳۹ و دیکشنری به ۸۵۵۶ رسید.

برای وارد کردن ملیت در کنار جملات، جملات غلط داده‌ها با [ملیت] الحاق شدند.

نتیجه تست گرفتن از این مدل با مجموعه داده تست به طول ۱۱۸۵ در جدول – آمده‌است.

۴-۳-۶-۳- مدل آموزش‌دیده همراه با برچسب‌های ملیت دسته‌بندی شده

به دنبال افزایش دقت مدل آموزش دیده با ملیت، چون داده‌های موجود از بعضی کشورها کم و اندک

بودند کشورهایی که ریشه زبان یکسانی دارند را در یک دست قرار دادیم که دسته‌های نهایی به صورت زیر می‌باشد:

- Arab:Bahrain','Algeria','Iraq','Kuwait','Lebanon','Palestine','Saudi Arabia','Sudan','Syria','Yemen
- Latin:'Argentina','Austria','Australia','Belgium','France','Colombia','Germany','Ghana','Hungary','Italy','IvoryCoast','Netherlands','Portugal','Romania','Senegal','Spain','Switzerland','United Kingdom','United States of America'
- Turk: 'Azerbaijan','Kazakhstan','Kyrgyzstan','Turkey'
- Pars: 'Afghanistan','Iran','Pakistan','Tajikistan'
- India: 'Bangladesh','India'
- Russia:'Belarus','Croatia','Poland','Russia','Serbia','Czech-Republic','Slovakia','Ukraine','Bulgaria','North Macedonia'
- China: China
- Other

به جای ملیت‌های اصلی فایل از این دسته ملیت‌ها برای آموزش استفاده کردیم و نتیجه ارزیابی مدل در جدول – آمده‌است.

۵-۳-۶-۳- مدل آموزش‌دیده بدون برچسب ملیت

این مدل همانند مدل آموزش دیده با ملیت، شامل داده‌های Form , Main_Sings می‌باشد. ورودی‌های این مدل فاقد ملیت می‌باشند. نتایج ارزیابی این مدل در جدول – نشان داده‌ایم.

۳-۷- مدل Error Detection

3-7-1- تحلیل کلی

این بخش از پروژه یک مدل یادگیری عمیق برای شناسایی خطاهای زبانی در متون فارسی با استفاده از مدل BERTرا پیاده‌سازی می‌کند. مراحل این فرآیند به شرح زیر است:

3-7-1- آماده‌سازی محیط و بارگذاری داده‌ها:

ابتدا کتابخلنه‌های مورد نیاز برای پردازش داده‌ها، ساخت و آموزش مدل‌های یادگیری عمیق و توکن‌سازی جملات بارگذاری می‌شوند. سپس فایل داده که شامل اطلاعات مختلفی درباره جملات و ویژگی‌های کاربران است، از Google Driveبارگذاری می‌شود.

3-7-2-پیش‌پردازش داده‌ها:

داده‌ها پاک‌سازی می‌شوند تا شامل تنها اطلاعات مورد نیاز برای مدل باشند. کاراکترهای اضافی و غیرضروری از جملات حذف می‌شوند تا کیفیت داده‌های ورودی به مدل بهبود یابد. این کار برای کاهش پیچیدگی و افزایش کارایی مدل ضروری است.

3-7-3-ترکیب داده‌ها و تقسیم‌بندی به مجموعه‌های آموزشی و آزمایشی:

جملات اشتباه با ملیت کاربران ترکیب می‌شوند تا مدل بتواند خطاها را با توجه به ملیت نیز تحلیل کند. سپس داده‌ها به صورت تصادفی به مجموعه‌های آموزشی و آزمایشی تقسیم می‌شوند.

3-7-3-1-توکن‌سازی و پد کردن جملات:

جملات به توکن‌هایی که نمایانگر کلمات یا کاراکترها هستند تبدیل می‌شوند. طول جملات یکسان‌سازی می‌شود تا تمامی ورودی‌ها به مدل طول یکسانی داشته باشند. این فرآیند کمک می‌کند تا مدل بتواند به صورت موثرتر و با دقت بیشتری جملات را تحلیل کند.

3-7-3-2-تعریف و اعمال برچسب‌ها:

برای هر جمله، برچسب‌هایی تعیین می‌شود که نشان می‌دهند آیا جمله خطایی دارد یا نه. این برچسب‌ها به مدل کمک می‌کنند تا خطاها را شناسایی کند. در اینجا برچسب‌ها به دو دسته اصلی تقسیم شده‌اند: درست (بدون خطا) و غلط (دارای خطا).

3-7-3-3-ساخت دیتاست و تقسیم‌بندی:

داده‌های آماده‌شده به یک دیتاست تبدیل می‌شوند و سپس به مجموعه‌های آموزشی و اعتبارسنجی تقسیم می‌شوند. این مجموعه‌ها برای آموزش مدل و ارزیابی عملکرد آن استفاده می‌شوند.

3-7-4-آموزش مدل:

مدل ParsBERT برای شناسایی خطاها آموزش داده می‌شود. در طول این فرآیند، مدل یاد می‌گیرد که چگونه جملات را تحلیل کند و خطاهای احتمالی را شناسایی نماید. پارامترهای مختلفی مانند تعداد اپوک‌ها (دوره‌های آموزشی)، نرخ یادگیری و اندازه بچ‌ها تنظیم می‌شوند تا مدل بهترین عملکرد را داشته باشد.

3-7-5-ارزیابی مدل:

پس از آموزش، مدل با استفاده از داده‌های آزمایشی ارزیابی می‌شود. مدل جملات را تحلیل کرده و خطاهای آنها را شناسایی می‌کند. دقت مدل با مقایسه نتایج پیش‌بینی‌شده با برچسب‌های واقعی سنجیده می‌شود.

3-7-6-پیش‌بینی و شناسایی خطاها:

در نهایت، مدل می‌تواند جملات جدید را تحلیل کرده و خطاهای موجود در آنها را شناسایی کند. این قابلیت برای کاربردهای مختلفی مانند تصحیح خودکار متن یا تشخیص خطاهای متنی در اسناد فارسی مفید است.

3-7-7-نتیجه‌گیری:

این کد یک فرآیند کامل و جامع برای شـــناســـایی خطاهای زبانی با اســـتفاده از مدل‌های یادگیری عمیق ارائه می‌دهد. اسـتفاده از مدل ParsBERT و تکنیک‌های پیش‌پردازش و توکن‌سـازی باعث شـده تا این مدل بتواند با دقت بالایی خطاهای متنی را در جملات فارسی شناسایی کند.

3-7-8-تحلیل و توضیح بخش دقت سنجی :

این کد برای تشخیص خطاهای زبانی در جملات فارسی با استفاده از مدل ParsBERTو انجام پردازش‌های لازم جهت محاسبه دقت مدل نوشته شده است. در ادامه، مراحل مختلف کد به تفصیل توضیح داده شده است:

3-7-8-1-بارگذاری کتابخانه‌ها:

در ابتدای کد، کتابخانه‌های مورد نیاز بارگذاری می‌شـــوند. این کتابخانه‌ها شـــامل ابزارهایی برای پردازش داده‌ها، استفاده از مدل‌های پیش‌ساخته، و محاسبه دقت مدل هستند. از جمله این کتابخانه‌ها می‌توان بهPandas ، Torch و Transformersاشاره کرد.

3-7-8-2-بارگذاری و پردازش داده‌ها:

فایل داده‌ها که شامل جملات و اطلاعات مرتبط است، از مسیر مشخص شده بارگذاری می‌شود. سپس ستون Error_Indexکه نشان‌دهنده محل خطاها در جملات اسـت، به کمک تابع Safe_Eval از حالت متنی به لیسـتی از اعداد تبدیل می‌شود. این تابع برای جلوگیری از بروز خطا در تبدیل رشته به لیست طراحی شده است.

در مرحله بعد، لیست‌های Error_Index به صورتی تغییر می‌کنند که اگر مقدار آنها غیر از صفر باشد، به عدد ۱ تبدیل شوند. این کار برای ساده‌سازی و آماده‌سازی داده‌ها برای پردازش مدل انجام می‌شود.

3-7-8-3-پیکربندی مدل:

در این مرحله، مقادیر id2label و label2id که نشان‌دهنده نگاشت برچسب‌ها به اعداد و بالعکس هستند، تنظیم می‌شوند. این مقادیر به مدل کمک می‌کنند تا برچسب‌های مناسب را به خروجی‌های خود تخصیص دهد.

3-7-8-4-تعریف تابع پیش‌بینی:

تابع Predict_Sentence برای انجام پیش‌بینی بر روی جملات ورودی تعریف شـده اسـت. این تابع مراحل زیر را طی می‌کند:

- ابتدا جمله ورودی توکنایز می‌شود (تبدیل به دنباله‌ای از کلمات/کاراکترها).
- سپس این توکن‌ها به مدل از پیش آموزش دیده ParsBERT داده می‌شوند تا آخرین لایه پنهان (Last Hidden State) استخراج شود.
- لایه‌ای خطی (Dense) به عنوان ســـر طبقه‌بندی تعریف شـــده و به آخرین لایه پنهان اعمال می‌شـــود تا پیش‌بینی‌های نهایی برای هر توکن به دست آید.

- با استفاده از تابع سیگموید، پیش‌بینی‌های عددی به مقادیر احتمالاتی تبدیل می‌شوند.
- در نهایت، با آستانه‌ای که تعیین شده است، این احتمالات به برچسب‌های ۰ یا ۱ (نمایانگر خطا یا عدم خطا) تبدیل می‌شوند.

3-7-8-5-پیش‌بینی خطاها

تابع Predict_Sentence به هر جمله در داده‌ها اعمال می‌شود تا پیش‌بینی‌های مدل برای هر جمله محاسبه شود. این پیش‌بینی‌ها در ستون Prediction ذخیره می‌شوند.

3-7-8-6-اصلاح پیش‌بینی‌ها

از آنجایی که توکن‌های ورودی ممکن است طول‌های مختلفی داشته باشند، تابع Trim_Predictions برای تطبیق طول پیش‌بینی‌ها با طول واقعی Error_Index تعریف شده است. این تابع پیش‌بینی‌ها را طوری اصلاح می‌کند که تنها پیش‌بینی‌های مربوط به محدوده خطاها نگه داشته شوند.

3-7-8-7-محاسبه دقت

تابع Compute_Accuracy برای محاسبه دقت مدل برای هر جمله تعریف شده است. این تابع، پیش‌بینی‌های اصلاح‌شده را با مقادیر واقعی خطا مقایسه می‌کند و دقت را برای هر جمله محاسبه می‌نماید. سپس میانگین دقت‌ها به عنوان دقت کلی مدل محاسبه می‌شود.

3-7-8-8-نمایش نتایج

در نهایت، دقت کلی مدل چاپ می‌شود که نشان‌دهنده عملکرد مدل در شناسایی خطاها در جملات فارسی است.

۳-۸- Web Application

برای پیاده‌سازی برنامه تحت وب از فریم ورک Django استفاده کردیم و از معماری MVT. این برنامه حاوی دو Application برای دو مدل Spell Correction و Error Detection میباشد که صفحه اصلی وارد محیطی می‌شویم که تصحیح خطا با استفاده از پنج مدل که ۳ تا از آنها را قبلا توضیح دادیم و ۲ تای دیگر بدون ملیت و بدون ارزیابی میباشند. (به دلیل کمبود داده‌های Train از تمامی داده‌ها برای آموزش استفاده کردیم که نتیجه بهتری را ارائه میدهد). با وارد شدن به Error_Detection/ وارد محیطی میشوید که مدل تشخیص خطا تعبیه شده است.

۳-۹- جمع‌بندی

در این فصل ابتدا صورت مسئله پروژه به بیان دیگر توضیح داده‌شد. سپس اطلاعات تکمیلی داده‌های استخراجی که شامل آمار و ارقام آنهاست در زیربخش سوم بیان شد و همچنین راهکاریی که برای پیش پردازش داده‌ها انجام دادیم در بخش بعدی آن توضیح دادیم. قبل از وارد شدن به مدل‌های سنگین هوش‌مصنوعی سعی کردیم داده‌ها را توسط الگوریتم‌های بدون نظارت دسته بندی کنیم و نتایج را ارزیابی کنیم که آیا نیاز به مدل‌های سنگین‌تر میشود یا خیر. سپس به توضیح کامل مدل‌های انتخابی هوش‌مصنوعی در دو عملیات Spell Correction و Error Detection پرداختیم. در نهایت از تکنولوژی که برای ساخت برنامه تحت وب استفاده کردیم صحبت کردیم و ساختار برنامه را مورد بررسی قرار دادیم.

فصل چهارم
نتایج

۴-۱- مقدمه

در این فصل به سنجش اطلاعات بدست آمده از مدل‌های آموزش در دو عملیات Spell Correction و Error Detection پرداخته می‌شود. نخست نتایج بدست آمده از مدل‌های Spell Correction و سپس Error بررسی شده است.

۴-۲- نتایج ارزیابی مدل Spell correction

جدول ۴-۱ نتایج ارزیابی مدل Spell correction

	Bleu Score	Precision	Recall	F1-Score	Accuracy
Without Nationality	0.59	0.74	0.71	0.70	0.71
With Nationality	0.55	0.68	0.65	0.64	0.65
With Batch Nationality	0.58	0.73	0.70	0.69	0.70

نتایجی که در جدول ۴-۱ آمده‌است، همگی توسط یک مدل با هایپرپارامتره یکسان بدست آمده‌است تنها تفاوتی که در اینجا حائز اهمیت است برچسب ملیت است که در هر سه این مدل‌ها متفاوت است.

زمانی که برچسب ملیت را به عنوان ورودی به مدل دادیم، انتظار می‌رفت که نتایج مدل بهبود یابند اما برخلاف تصورات ما اینطور پیش نرفت. همچنین متوجه می‌شویم که زمانی که ملیت‌ها را دسته بندی میکنیم، نتایج آموزش بهتر می‌شود. می‌توان از این ارقام این نتیجه را گرفت که تعداد داده‌ها کم است؛ زیرا زمانی که

هر ملیت به عنوان عضو جداگانه برچسب دهی می‌شود، مدل حتی در تلاش برای یادگیری ملیت‌هایی است که از آنها فقط ۳ داده وجود دارد و با دسته‌بندی کردن ملیت‌ها، داده‌های با برچسب‌های اندک حذف می‌شوند و مدل پیشرفت میکند. همچنین این نکته نیز حائز اهمیت است که با جدا کردن دیتاست Train از Tset تعداد کل داده‌های Train به ۴۷۳۹ عدد می‌رسد که ۲۰ درصد آن نیز هم برای دیتاست Validation جدا می‌شود یعنی در نهایت می‌شود ۳۷۹۱ عدد داده که این عدد در عملیات‌های سنگینی مثل پردازش زبان عدد کمی به شمار می‌آید. چه بسا مدل در تلاش برای یادگیری رابطه ملیت با غلط‌های املایی می‌باشد اما با کمبود داده مواجه می‌شود.

در نتیجه ارزیابی توسط معیار Bleu Score، عدد بدست آمده نسبتا خوب به‌نظر می‌رسد اما باید به این توجه داشت که ما در یک دنباله از کلمات به دنبال تصحیح خطای فقط یک کلمه میگردیم که برای این عملیات عدد کمی است چه بسا انتظار ما باید بیشتر از این باشد زیرا مابقی کلمات یکسان هستند.

۴-۳- نتایج ارزیابی مدل Error Detection

این قسمت به یک فرآیند کامل از بارگذاری داده‌ها، پیش‌پردازش، پیش‌بینی، و ارزیابی دقت مدل برای شناسایی خطاهای زبانی را پیاده‌سازی می‌کند اشاره دارد. استفاده از مدل Parsbert و روش‌های طبقه‌بندی توکنی، به دقت نسبتا مناسبی مدل در شناسایی خطاهای متنی کمک کرده است.

جدول ۳- نتایج ارزیابی ۳مدل Error detection

	مدل با ترکیب ملیت های مشابه	مدل آموزش دیده بدون ملیت	مدل آموزش دیده با ملیت
دقت	۰.۶	۰.۵۵	۰.۶۵

۴-۴- جمع‌بندی

در این فصل به نتایج حاصل از مدل‌های آموزش دیده با داده‌هایمان پرداختیم و تحلیل کردیم. همانطور که متوجه شدید، نتایج مدل‌ها مطابق پیش‌بینی و انتظارات ما پیش نرفت. دلایل متعددی می‌توانند در این امر دخیل باشند از جمله تعداد داده محدود و کم، جملاتی با طول بسیار بلند، متفاوت بودن جملات نوشته شده توسط زبان‌آموزان با ملیت‌های متفاوت، تعداد ملیت خیلی زیاد در مقابل تعداد داده اندک.

فصل پنجم
نتیجه‌گیری و پیشنهادها

۵-۱-محدودیت‌ها

در این پروژه ما با محدودیت‌های متفاوتی مواجه بودیم. به عنوان اولین محدودیت می‌توان به تعداد داده کم اشاره کرد. این مجموعه با پاکسازی و جدا کردن Test , Validation در مجموع نزدیک به ۴۷۰۰ داده برای Train باقی می‌گذارد که در عملیات پردازش متن مخصوصا در زبان فارسی که جز زبان‌های پیچیده محسوب می‌شود، به طور قابل توجه‌ای کم می‌باشد.

محدودیت دیگری که می‌توان به آن اشاره کرد، تفاوت فایل‌های نوشته شده، توسط ملیت‌های گوناگون است. بهتر بود که تمامی ملیت‌ها یک فایل یکسان را می‌نوشتند در نتیجه تفاوت غلط‌های املائی در این مجموعه داده، قابل استنتادتر می‌بود.

مشکل دیگری که در مجموعه داده، وجود دارد این است که اندازه جمله‌هایی که توسط مصححان انتخاب شده‌است، زیاد است. برای این کار تصحیح و تشخیص خطا، بهتر است یا طول ورودی کم باشد یا اگر زیاد است تعداد غلط‌های املایی بیشتر از یکی باشد. ما در این مجموعه داده فایل‌هایی را مشاهده کردیم که در یک لیست توکن بندی شده از کلمات به طول ۲۰۹ فقط ۱ کلمه به عنوان اشتباه برچسب شده، این امر باعث می‌شود عملکرد و تشخیص برای Attention ها سخت و دشوار شود.

چون مدل انتخابی برای آموزش، مدل سنگینی است، نیازمند شدیم که در بستر وب از google colab استفاده کنیم که به ما ۱۵ گیگابایت کارت گرافیک تخصیص داد استفاده کنیم. متاسفانه محدودیت زمانی استفاده برای این منابع وجود دارد و به مدت ۳ ساعت در روز بیشتر برای ما امکان استفاده فراهم نمی‌کند.

۵-۲- نتیجه‌گیری

نتیجه‌گیری روی نتایج مدل‌های بدست آمده به دلایل گفته شده، یک مقدار کار دشواری می‌باشد. در

کل مدل آموزش دیده، پتانسیل و توانایی یادگیری بهتر توسط ملیت را از خود نشان داد از آنجایی که ملیت‌ها را دسته‌بندی کردیم اما نتوانست از مدل بدون ملیت نتایج بهتری بدست بیاورد تقریبا به همان اندازه، نتیجه گرفت. در نتیجه می‌توان امیدوار بود که با افزایش داده و برطرف کردن یکسری مشکل مدلی که با ملیت آموزش ببیند، نتیجه بهتری نسبت به مدل بدون ملیت آموزش دیده، ارائه دهد.

۳-۵- پیشنهادها

پیشنهاد می‌شود در کنار این آموزش با ملیت‌های گوناگون ابتدا یک دیکشنری با سایز بزرگ تهیه شود و در مدل از آن استفاده شود و کلمات دیتاست به آن اضافه شوند. تحت این صورت مدل دیگر شناخت جامع‌تر و دایرالغات گسترده‌تری در زبان فارسی دارد. این امر موجب می‌شود با کلماتی که در دیتاست اولیه موجود نیست و در حالت عادی به صورت "<UNK>" معرفی میشد را بشناسد و بردار مخصوص به آن را توسط لایه embedding تهیه کند.

پیوست ۱: لیست برنامه‌ها

اکثر برنامه‌هایی که برای این پروژه استفاده شده‌اند در فرمت ipynb. می‌باشند به جز فایل‌هایی که برای برنامه تحت وب استفاده می‌شود. لیست برنامه‌ها به صورت زیر است:

- Unzip.Ipynb : برنامه‌ای که برای باز کردن کل فایل‌های زیپ شده داده‌ها استفاده شد.
- Data_Extraction.Ipynb : استخراج داده‌ها از فایل‌های Xmi.
- Fixing-Nationality-Feature.Ipynb : مرتب سازی داده‌ها بر اساس حروف الفبا ستون ملیت
- Modifying_Tashdid_Data.Ipynb : تغییر تشدید به عنوان زیر مقوله [۲]
- Nanvalue.Ipynb : حذف و آنالیز داده‌های که مقدار Nan دارند.
- Analysis.Ipynb : آنالیز و گرفتن آمار داده‌های موجود
- Clustering.Ipynb : دسته‌بندی‌ها با الگوریتم بدون نظارت و ارزیابی آنها
- Data_Analysis.Ipynb : تولید کردن تصاویر و نمودارهای آنالیز داده
- All_Model.Ipynb : تولید مدل Spell Correction از همه داده‌ها با احتساب جداسازی Validation . این فایل در Google Colab اجرا شده‌است.
- All_Without_Validation.Ipynb : تولید مدل Spell Correction از همه داده‌ها بدون احتساب جداسازی Validation . این فایل در Google Colab اجرا شده‌است.
- With_Nationality_Model.Ipynb : تولید مدل Spell Correction از داده‌های Main_Category و Form با احتساب جداسازی Validation و Test و برچسب ملیت به عنوان ورودی. این فایل در Google Colab اجرا شده‌است.
- With_Batch_Nationality.Ipynb : تولید مدل Spell Correction از داده‌های main_category و Form با احتساب جداسازی Validation و Test و برچسب دسته ملیت به عنوان ورودی. این فایل در Google Colab اجرا شده‌است.
- Without_Nationality_Model.Ipynb : تولید مدل Spell Correction از داده‌های Main_Category و Form با احتساب جداسازی Validation و Test . این فایل در Google Colab اجرا شده‌است.
- Scores_With_Nationality: ارزیابی مدل مربوطه
- Scores_With_Nationality: ارزیابی مدل مربوطه
- Scores_With_Nationality: ارزیابی مدل مربوطه
- All_Model.Keras : مدل ذخیره شده مربوط به کل داده‌ها با احتساب جداسازی Validation
- All_Without_Validation.Keras : مدل ذخیره شده مربوط به کل داده‌ها بدون احتساب

جداسازی Validation

- With_Nationality_Model.Keras : مدل ذخیره شده مربوط به مدل
- With_Batch_Nationality.Keras : مدل ذخیره شده مربوط به مدل
- Without_Nationality_Model.Keras: مدل ذخیره شده مربوط به مدل
- Data.Csv : فایل داده‌های استخراج شده که اجازه پخش آن را نداریم.
- Metadata.Csv : فایل اطلاعات تکمیلی هر زبان‌آموز
- Analysis.Csv: فایل آمار و ارقام حاصل از آنالیز داده‌ها

منابع:

[1] INCEpTION Project, "INCEpTION - Semantic Annotation Platform." [Online]. Available: https://inception-project.github.io/. [Accessed: August, 30 2024].

[2] J. L. Elman, "Finding Structure in Time," Cognitive Science, vol. 14, no. 2, pp. 179-211, 1990.

[3] A. Vaswani et al., "Attention is all you need," in Advances in Neural Information Processing Systems, 2017, pp. 5998–6008.

[4] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997

[5] A. Radford et al., "Language Models are Unsupervised Multitask Learners," OpenAI, 2019.

[6] A. Conneau et al., "Unsupervised Cross-lingual Representation Learning at Scale," in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 8440-8451.

[7] L. Xue et al., "mT5: A Massively Multilingual Pre-trained Text-to-text Transformer," in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2021, pp. 483-498. [8] ParsBert

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," arXiv preprint arXiv:1301.3781, 2013.

[10] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532-1543.

[11] Google, "Google Colaboratory." [Online]. Available: https://colab.research.google.com/. [Accessed: August 30, 2024].