پروژه پایانی بوت‌کمپ یادگیری ماشین

مریم بیابانی، نرگس حاج ملاعلی، محمدامین یادگاری، عاطفه امانی، سحر نقدی

ML-Gladiators

مرداد- مهر ۱۴۰۳

مقدمه:

در این پروژه، تمرکز ما بر توسعه یک سیستم ترجمه ماشینی برای تبدیل زیرنویس‌های انگلیسی به فارسی است. با توجه به اهمیت زیرنویس‌ها در انتقال مفهوم فیلم‌ها به زبان‌های مختلف، ایجاد یک سیستم ترجمه دقیق و کارآمد ضروری است. استفاده از مدل‌های پیشرفته ترجمه ماشینی و همچنین ارزیابی دقیق عملکرد این مدل‌ها، به ما کمک می‌کند تا سیستمی با کیفیت بالا ارائه دهیم. این گزارش به بررسی گام‌های مختلف در فرآیند ساخت این سیستم و نتایج حاصل از ارزیابی آن می‌پردازد.

این پروژه در چند گام انجام شده است که شامل موارد زیر است:

- **جمع‌آوری داده‌های مورد نیاز:** داده‌های لازم برای آموزش سیستم از منابع مختلف جمع‌آوری شده است.
- **پیش‌پردازش داده‌ها:** داده‌ها قبل از استفاده، برای بهبود کیفیت و کارایی ترجمه، پیش‌پردازش شده‌اند.
- **ایجاد سیستم‌های ترجمه مبتنی بر مدل‌های غیر LLM:** در این بخش، از مدل‌های RNN و LSTM استفاده شده است تا سیستم‌های اولیه ترجمه ایجاد شوند. به علاوه برای مدل‌های مبتنی بر Sequence to Sequence، مدل‌هایی مانند BART و مدل تنظیم‌شده T5 پیاده‌سازی شده‌اند که عملکرد بهتری نسبت به مدل‌های قبلی ارائه می‌دهند.
- **ایجاد سیستم‌های ترجمه مبتنی بر LLM:** در این گام، سیستم‌های مبتنی بر مدل‌های زبانی عظیم (LLM) توسعه یافته‌اند تا دقت و کارایی ترجمه بهبود یابد.
- **ارزیابی مدل‌های پیاده‌سازی شده:** با استفاده از امتیاز BLEU و داده‌های تست جمع‌آوری شده، مدل‌هایی که پیاده‌سازی شده است، ارزیابی شدند.
- **داکرایز کردن پروژه:** در نهایت، مدل T5 تنظیم شده، داکرایز شد.

در پایان، نتیجه‌گیری و پیشنهاداتی برای توسعه‌های آتی بیان شده است.

گام صفر - جمع‌آوری داده‌های مورد نیاز

در این مرحله، داده‌های مورد نیاز برای آموزش سیستم ترجمه به دو روش جمع‌آوری شد:

1. **استفاده از مجموعه داده‌های آماده:** در این روش، از داده‌های موجود در سایت‌ها و مقالات مختلف استفاده شد که شامل مجموعه‌های زیر است:
   - مجموعه داده میزان: این مجموعه یک پیکره موازی فارسی-انگلیسی است که شامل حدود ۱ میلیون جفت جمله از آثار برجسته ادبیات است.
   - مجموعه داده PEPC: شامل جملات موازی انگلیسی و فارسی است که از طریق یک روش ترجمه دوطرفه از اسناد ویکی‌پدیا استخراج شده‌اند.
   - مجموعه داده TEP: این مجموعه، پیکره‌ای موازی انگلیسی-فارسی است که توسط آزمایشگاه پردازش زبان طبیعی و متن دانشگاه تهران ارائه شده و به‌عنوان اولین پیکره آزاد انگلیسی-فارسی شناخته می‌شود.
2. **استخراج زیرنویس‌ها فیلم‌ها از سایت‌ها مرتبط:** در این روش، زیرنویس‌های فارسی و انگلیسی فیلم‌های "Soul"، "Lifeguard"، "Inside Out 2"، "Detained" و "Twister" از سایت sub2fm با استفاده از یک کد پایتون استخراج شدند. سپس، با استفاده از کتابخانه pysrt، زیرنویس‌های فارسی و انگلیسی بر اساس زمان‌بندی فایل‌های SRT تطبیق داده شده و جملات متناظر گردآوری شدند.

در نهایت، تمامی داده‌های جمع‌آوری‌شده، شامل حدود ۲ میلیون ردیف، در یک فایل CSV ذخیره شدند. البته لازم به ذکر است که به دلیل چالش‌های مربوط به آموزش مدل‌ها با چنین حجم بزرگی از داده، برای مراحل بعدی پروژه، به‌صورت تصادفی حدود ۷۰۰۰ داده از این مجموعه انتخاب شد و مورد استفاده قرار گرفت.

گام اول - پیش‌پردازش داده‌ها

هدف اصلی در این گام، تمیز کردن و آماده‌سازی داده‌ها برای استفاده در مراحل بعدی بود. پیش‌پردازش داده‌ها در تسک ترجمه انگلیسی به فارسی اهمیت زیادی دارد، زیرا داده‌های خام معمولاً شامل نویزهایی مانند علائم نگارشی نامناسب، فاصله‌های اضافی و ساختارهای غیر معمول هستند که می‌توانند دقت مدل‌های ترجمه را کاهش دهند.

همچنین، در متون فارسی وجود حروف عربی و استفاده نادرست از فاصله به جای نیم‌فاصله از چالش‌های اصلی پیش‌پردازش محسوب می‌شوند. این مشکلات اگر به‌درستی حل نشوند، می‌توانند منجر به اختلال در فهم مدل و کاهش کیفیت خروجی شوند. پیش‌پردازش دقیق داده‌ها به مدل کمک می‌کند تا ورودی‌ها را بهتر تحلیل کرده و ترجمه‌های دقیق‌تر و باکیفیت‌تری ارائه دهد.

در نتیجه، در این گام، عملیات پیش‌پردازش برای داده‌های انگلیسی و فارسی به‌صورت جداگانه انجام شد تا کیفیت نهایی داده‌ها برای آموزش مدل‌های ترجمه بهینه شود.

برای پیش پردازش داده‌های زبان فارسی عملیات‌های زیر صورت گرفت:

- حذف تگ‌های HTML که برای نمایش رنگی زیرنویس‌ها استفاده می‌شدند با استفاده از کتابخانه regex در پایتون.
- نرمال‌سازی متون فارسی به کمک کتابخانه hazm که شامل موارد زیر است:
  - اصلاح فاصله‌گذاری‌ها در متن، نشانه‌های سجاوندی و پیشوندها و پسوندها.
  - حذف اعرابِ حروف.
  - حذف برخی از کاراکترها و نشانه‌های خاص که کاربردی در پردازش متن ندارند.
  - کاهش تکرارهای بیش از دو بار حروف به دو بار. (مثلاً تبدیل «سلاممم» به «سلامم»)
  - انجام اصلاحات مخصوص زبان فارسی. (مثلاً جایگزین‌کردن کوتیشن با گیومه)
  - تبدیل ارقام انگلیسی به ارقام فارسی.
  - جایگزین کردن کاراکترهای یونیکد با معادل نرمال‌شده‌ی آن.
  - جداسازی پیشوند «می» و «نمی» در افعال.
  - بررسی غلط‌های املایی در متون فارسی به کمک کتابخانه dadmatools.
- جایگذاری حروف عربی موجود در متون با معادل فارسی آن و یکسان‌سازی حروفی که معنای مشابهی دارند ولی به دلیل استفاده از کیبوردهای مختلف، به‌صورت متفاوت نوشته شده‌اند. این حروف شامل موارد زیر بود:
  - «آ»،«إ»،«أ» که همگی به «ا» تبدیل شدند.
  - «ي»،«ئ» که هر دو به «ی» تبدیل شدند.

- «ؤ» به «و» تبدیل شد.
- «ة» به «ه» تبدیل شد.
- «ك» به «ک» تبدیل شد.

برای پیش پردازش داده‌های زبان انگلیسی عملیات‌های زیر صورت گرفت:

- حذف تگ‌های HTML که برای نمایش رنگی زیرنویس‌ها استفاده می‌شدند با استفاده از کتابخانه regex در پایتون.
- نرمال‌سازی متون انگلیسی به کمک کتابخانه unicodedata که شامل موارد زیر است:
  - تبدیل رشته به فرمت نرمال‌سازی Unicode به شکل (NFD). در این حالت، کاراکترهای ترکیبی (مثل حروفی که با اعراب یا اکسان نوشته شده‌اند) به اجزای ساده‌ترشان شکسته می‌شوند. (مثلاً حرف "é" به دو بخش "e" و "´" تقسیم می‌شود.)
  - تبدیل رشته نرمال‌شده به فرمت ASCII و حذف کاراکترهای غیر-ASCII (مثل اعراب‌ها یا کاراکترهای خاص).
  - تبدیل بایت‌ها از فرمت UTF-8 به رشته متنی.
- کوچک کردن همه حروف انگلیسی و حذف فاصله‌های اضافی به کمک کتابخانه regex در پایتون.
- بررسی غلط‌های املایی در متون انگلیسی به کمک کتابخانه spellchecker در پایتون.

علاوه بر پیش‌پردازش‌های خاص زبان، ردیف‌هایی که یکی از متن‌های فارسی یا انگلیسی در آن‌ها خالی بود، حذف شدند. در پیش‌پردازش داده، از سایر عملیات‌های رایج در تسک‌های متنی، مانند حذف علائم نگارشی، حذف Stop word و عملیات‌های Stemming و Lemmatization، به دلیل اهمیت معنایی متون در ترجمه و جلوگیری از دست دادن بار معنایی آن‌ها، استفاده نشد.

در نهایت، تمام مجموعه داده را به طور تصادفی جابجا (Shuffle) کرده و به دو مجموعه آموزش و تست تقسیم‌بندی کردیم. مجموعه تست شامل ۱۰۰۰ ردیف و مجموعه آموزش حدود ۶۰۰۰ ردیف را داراست.

گام دوم - سیستم ترجمه مبتنی بر مدل‌های غیر LLM:

در گام دوم پروژه به پیاده سازی سیستم ترجمه ماشینی با رویکردهای کلاسیک پرداختیم و با مدل‌های RNN، LSTM و Seq2Seq سیستم‌های ترجمه پیاده‌سازی کردیم. در ادامه‌ی این بخش، هر کدام از رویکردها به طور کامل شرح داده شده است.

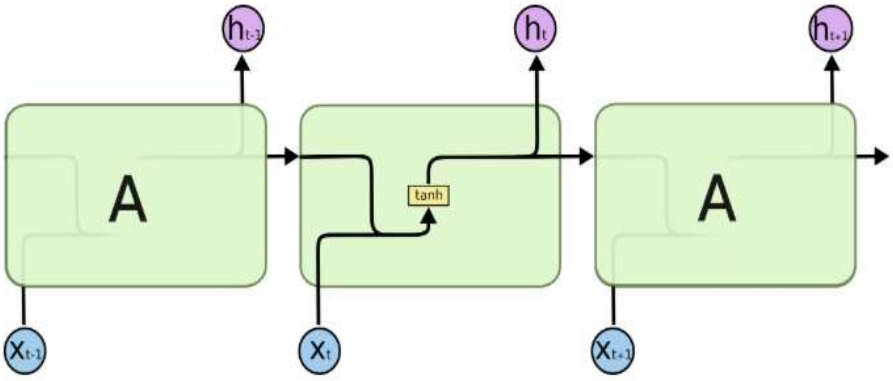
شبکه‌های عصبی بازگشتی(RNN):

شبکه‌های عصبی بازگشتی (RNN) نوعی شبکه عصبی هستند که به طور خاص برای داده‌های ترتیبی طراحی شده‌اند. ویژگی اصلی RNNها این است که دارای حافظه داخلی هستند که اطلاعات مربوط به حالت‌های قبلی را ذخیره می‌کند. این حافظه باعث می‌شود تا RNN بتواند اطلاعات پیشین را در تصمیم‌گیری‌های فعلی در نظر بگیرد.

در یک RNN، برخلاف شبکه‌های عصبی معمولی که ورودی‌های خود را به صورت مستقل از یکدیگر پردازش می‌کنند، هر واحد از شبکه می‌تواند اطلاعاتی از مراحل قبلی را برای مراحل بعدی به اشتراک بگذارد. این قابلیت باعث می‌شود که RNNها برای وظایفی مانند پیش‌بینی سری‌های زمانی، تشخیص الگوها در داده‌های ترتیبی و پردازش زبان طبیعی (NLP) بسیار کارآمد باشند.

علی‌رغم مزایای RNN، این شبکه‌ها با مشکلاتی همچون محوشدگی گرادیان و انفجار گرادیان مواجه‌اند. این مشکلات به دلیل زنجیره طولانی محاسبات در طول زمان رخ می‌دهد و باعث می‌شود که در فرآیند یادگیری، مقادیر گرادیان‌ها به مرور زمان کوچک یا بسیار بزرگ شوند. در نتیجه، مدل نمی‌تواند به درستی پارامترهای مربوط به اطلاعات دورتر را یاد بگیرد. به همین دلیل، RNN ها در به‌خاطر سپردن وابستگی‌های طولانی‌مدت ناکارآمد هستند.

در شکلی که در ادامه آمده است، ساختار کلی یک شبکه RNN قابل مشاهده است.



ما در این پروژه، با استفاده از شبکه‌های RNN یک سیستم ترجمه پیاده‌سازی کردم. مراحلی که برای پیاده‌سازی این سیستم صورت گرفت، به شرح زیر است:

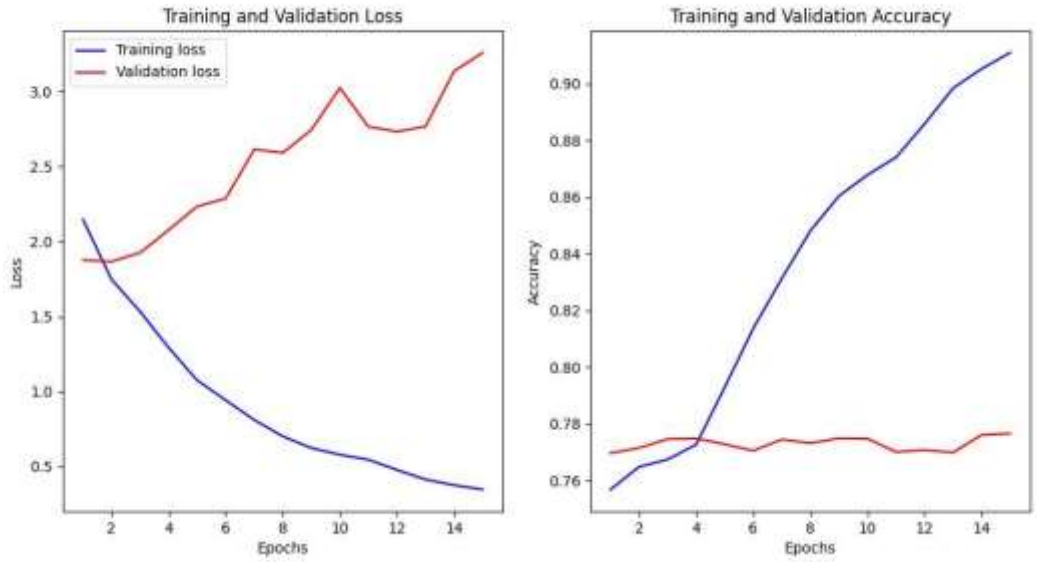
1. **بارگذاری داده‌ها:** داده‌های آموزشی و تست از دو فایل CSV حاوی جملات انگلیسی و ترجمه فارسی آن‌ها بارگذاری شدند. مجموعه آموزشی شامل جملات انگلیسی با ترجمه‌های فارسی است و مجموعه تست فقط جملات انگلیسی دارد.
2. **استخراج ویژگی‌ها با BERT:** از مدل BERT چندزبانه (Multilingual BERT) استفاده شد تا جملات انگلیسی به بردارهای ویژگی (یا تعبیه‌های متنی) تبدیل شوند. این بردارها شامل اطلاعات معنایی هستند و به عنوان ورودی به مدل RNN داده می‌شوند. هر جمله انگلیسی پس از توکن‌سازی توسط BERT به برداری با طول ثابت ۲۰ کلمه و ویژگی‌های معنایی تبدیل شد.
3. **پیش‌پردازش جملات فارسی:** جملات فارسی توکن‌سازی شده و به توالی‌های عددی تبدیل شدند که هر عدد نمایانگر یک کلمه در واژگان است. این توالی‌ها با استفاده از padding به طول ثابت تنظیم شدند تا ورودی‌های مدل دارای اندازه یکسانی باشند.
4. **ساخت مدل RNN:** از یک شبکه SimpleRNN دوطرفه با سه لایه RNN استفاده شد. این لایه‌ها توانایی پردازش توالی‌های داده از دو جهت را دارند و به مدل کمک می‌کنند تا اطلاعات بهتری از ترتیب کلمات استخراج کند. هر لایه RNN دارای تعدادی نورون است (۵۱۲ و ۲۵۶) و تابع فعال‌سازی tanh برای پردازش داده‌های متوالی استفاده شد. برای جلوگیری از بیش‌برازش (Overfitting)، لایه‌های Dropout بعد از هر

لایه RNN اضافه شدند. لایه نهایی مدل یک Dense layer با تابع فعال‌سازی Softmax است که توالی کلمات فارسی را به عنوان خروجی تولید می‌کند.

5. **آموزش و ارزیابی مدل:** مدل با استفاده از داده‌های آموزشی برای ۱۵ دوره (epoch) آموزش داده شد. داده‌های اعتبارسنجی (Validation) نیز برای ارزیابی عملکرد مدل در طول آموزش استفاده شد. همچنین، تابع هزینه Sparse Categorical Crossentropy برای به حداقل رساندن خطای پیش‌بینی‌ها و بهینه‌ساز Adam برای بهبود سرعت یادگیری استفاده شدند.
6. **ترجمه و ارزیابی جملات تست:** پس از آموزش مدل، جملات انگلیسی در مجموعه تست به فارسی ترجمه شدند. فرآیند ترجمه شامل تبدیل جملات انگلیسی به بردارهای ویژگی با BERT و سپس استفاده از مدل RNN برای تولید توالی ترجمه فارسی است. در نهایت، نتایج ترجمه نهایی در فایل CSV ذخیره شدند.

برای پیاده‌سازی چنین سیستمی به کمک مدل‌های RNN چندین مرحله طی شد و به مدل‌های مختلفی رسیدیم. در نهایت، با بررسی نتایج این مدل‌ها، بهترین مدل انتخاب شد. در ادامه، مراحلی که طی شد تا به بهترین مدل برسیم در ادامه آورده شده است.

برای یافتن بهترین مدل برای پیاده سازی سیستم ترجمه با RNN، ابتدا مدلی طراحی کردیم که یک لایه SimpleRNN با ۵۱۲ نورون را دارا بود. برای تابع فعالسازی از ReLu و Dropout با مقدار ۰.۲ استفاده شد. برای این مدل، در نهایت، مقدار val_loss مساوی ۳.۲۵ و مقدار val_accuracy مساوی ۰.۷۷ حاصل شد. همانطور که در نموداری که در ادامه آمده است، مشخص است اندازه loss مدل خیلی زیاد شد و accuracy هم به نسبت زیاد نیست.



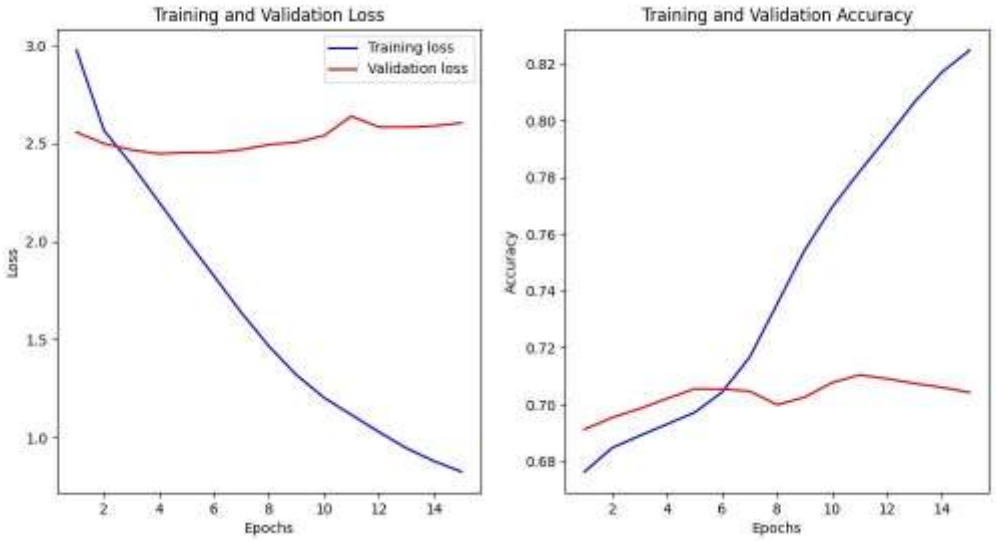
در ادامه، با توجه به اینکه در پیاده سازی شبکه های عصبی بازگشتی (RNN) تابع فعال سازی tanh از ReLU بهتر است، برای مدل دوم از دو لایه RNN دوطرفه استفاده شد که لایه اول دارای ۵۱۲ نورون و لایه دوم دارای ۲۵۶ نورون داشت. همچنین برای جلوگیری از overfitting مقدار Dropout را افزایش و برابر ۰.۵ قرار دادیم و از تابع فعال سازی tanh استفاده کردیم. در پایان، مقدار خطای این مدل نسبت به مدل قبل کاهش یافت و به مقدار ۲.۷۱ رسید ولی قابل ذکر است که در این مدل، دقت مدل هم کاهش یافت و مقدار ۰.۶۹ حاصل شد.

در اینجا لازم است که اشاره کنیم که دلیل استفاده از تابع فعال سازی tanh به جای ReLU این است که در فرآیند ترجمه ماشینی، مدل باید ظرایف و جزییات کلمات و عبارات را در نظر بگیرد و ترجمه مناسبی ارائه دهد. استفاده از tanh به دلیل فعال‌سازی پیوسته و مشتق‌پذیری خوب آن، به شبکه امکان می‌دهد تغییرات نرم و پیوسته‌ای در طول یادگیری داشته باشد و به‌طور ملایم‌تری وابستگی‌های زمانی را مدیریت کند. تابع ReLU بیشتر مناسب برای کاربردهایی است که به مقادیر خروجی بزرگ و یادگیری سریع نیاز دارند، اما در ترجمه ماشینی که نیاز به ظرافت بیشتری وجود دارد، تابع tanh عملکرد بهتری از خود نشان می‌دهد.

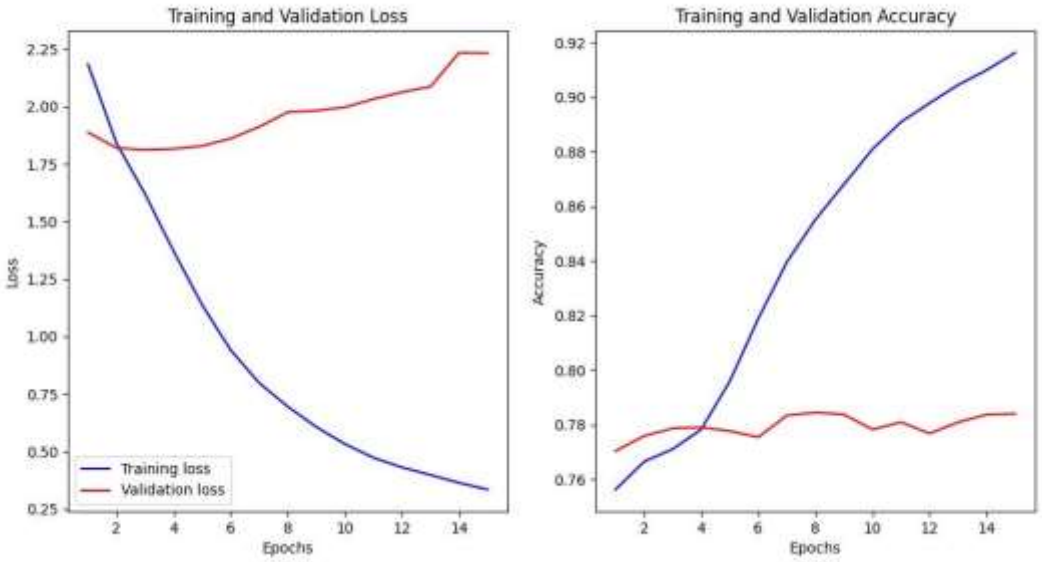
به علاوه، دلیل استفاده از RNN دوطرفه این بود که RNN دوطرفه با پردازش دنباله‌ها به طور همزمان از دو سمت (چپ به راست و راست به چپ) این امکان را فراهم می‌کند که مدل به اطلاعات پیشین و آینده هر کلمه دسترسی داشته باشد. این ویژگی به درک بهتر متن کمک می‌کند. همچنین، با استفاده از اطلاعات دو طرفه، مدل قادر است

الگوها و روابط پیچیده‌تری را شناسایی کند، که به بهبود دقت پیش‌بینی‌ها در ترجمه کمک می‌کند. هم چنین RNN دوطرفه می‌تواند وابستگی‌های طولانی‌مدت را بهتر مدیریت کند، زیرا اطلاعات از دو سمت به هم متصل می‌شوند و از ناپایداری‌هایی که ممکن است در RNNهای تک طرفه ایجاد شود، جلوگیری می‌کند. در مجموع، RNN دوطرفه به بهبود کیفیت ترجمه و تحلیل متون کمک شایانی می‌کند.

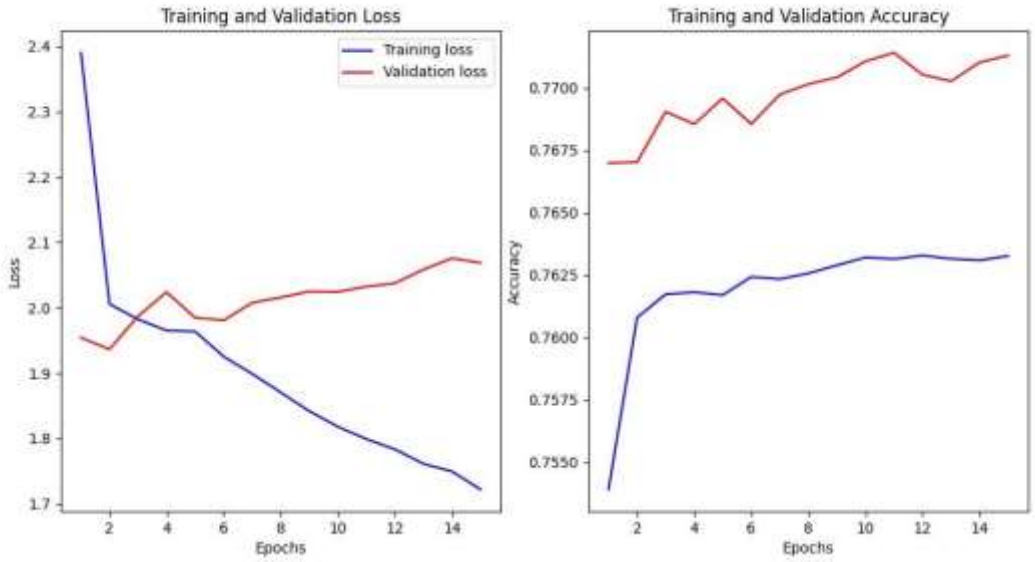
در ادامه نمودار مدل دوم آمده است. همانطور که در نمودار مشخص است، در این مدل، هم چنان loss مدل زیاد و accuracy مدل پایین است. با این حال، این مدل نسبت به مدل اول بهبود پیدا کرد و مقدار val_loss برابر با ۲.۷ و مقدار val_accuracy برابر با ۰.۶۹ حاصل شد. البته شایان ذکر است که val_accuracy نسبت به مدل قبل کمی کاهش داشت و از مقدار ۰.۷۷ به مقدار ۰.۶۹ رسید، ولی در این مدل توانستیم که مقدار val_loss مدل را کاهش دهیم.



پس از دو مدل قبل، مدل سومی مبتنی بر شبکه RNN پیاده‌سازی کردیم. در این مدل، از یک لایه RNN دوطرفه با ۵۱۲ نورون و تابع فعال سازی tanh و Dropout با مقدار ۰.۵ استفاده کردیم. در این مدل نتایج بهتری نسبت به دو مدل قبل به دست آوردیم؛ مقدار خطای مدل نسبت به دو مدل قبل کاهش یافت و مقدار ۲.۲۳ حاصل شد. همچنین دقت هم افزایش یافت و به ۰.۷۸ رسید که نسبت به دو مدل قبل خیلی بهتر بود. نمودار متناظر با این مدل در ادامه آمده است.



در نهایت، به مدلی رسیدیم که به عنوان بهترین مدل ما انتخاب شد. این مدل سه لایه RNN دوطرفه دارد که لایه اول ۵۱۲ نورون، لایه دوم ۲۵۶ نورون و لایه سوم نیز ۲۵۶ نورون دارد. برای تابع فعال‌سازی این مدل، از تابع tanh استفاده شد و همچنین مقدار Dropout هم برابر با ۰.۵ قرار دادیم. مقدار خطای این مدل نسبت به سه مدل قبل با کاهش خوبی به مقدار ۲.۰۶ رسید و همچنین دقتی که برای این مدل به دست آمد برابر با ۰.۷۷ است. در نهایت، با مقادیر به دست آمده، مشخص است که این مدل نسبت به مدل های قبلی بهتر عمل می‌کند. با این حال، خوب است که اشاره کنیم که شبکه‌های RNN، هنگام یادگیری وابستگی‌های طولانی‌مدت بین کلمات، با مشکل محوشدگی گرادیان روبرو هستند. این مشکل باعث می‌شود که مدل نتواند اطلاعات مهمی را که در ابتدای یک جمله یا متن آمده است، به‌خوبی به خاطر بسپارد و از آن در ترجمه استفاده کند. همچنین در جملات طولانی، RNN نمی‌تواند به خوبی ارتباط کلمات ابتدایی با کلمات انتهایی را حفظ کند، که این موضوع در ترجمه ماشینی، جایی که حفظ معنا در طول جمله حیاتی است، مشکل‌ساز می‌شود. بنابراین هر چند که مدل نهایی ما دقت بالایی داشت (۰.۷۷) و مقدار خطای آن نیز کم بود (۲.۰۶) اما برای ترجمه خیلی عملکرد خوبی نداشت و مدل ما دچار overfitting شد. در ادامه، نمودار مدل نهایی ما به شکل زیر است:

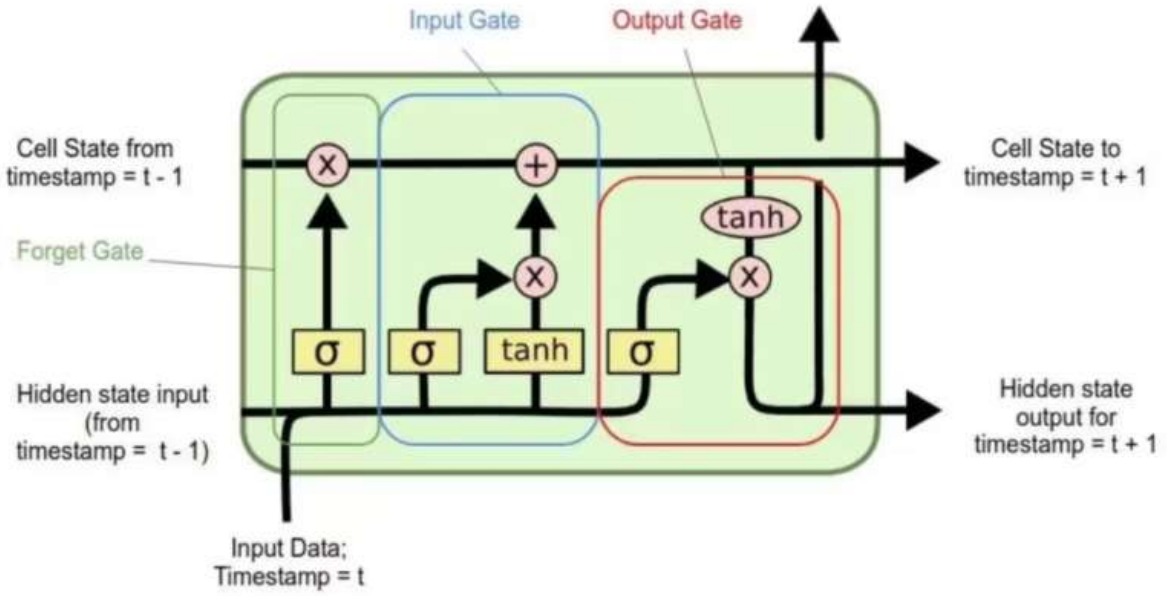


شبکه های حافظه طولانی کوتاه‌مدت (LSTM):

شبکه حافظه طولانی کوتاه‌مدت (LSTM) نوع بهبود یافته شبکه RNN است که برای رفع مشکلات ناپایداری گرادیان‌ها طراحی شده است. شبکه LSTM دارای ساختار سلولی خاصی است که شامل سه دروازه اصلی است:

1. **دروازه ورودی (Input Gate):** این دروازه مشخص می‌کند که چه مقدار از اطلاعات جدید وارد حافظه سلولی شود.
2. **دروازه فراموشی (Forget Gate):** این دروازه مشخص می‌کند که چه مقدار از اطلاعات قبلی باید حذف شود.
3. **دروازه خروجی (Output Gate):** این دروازه خروجی نهایی را براساس وضعیت حافظه و ورودی فعلی مشخص می‌کند.

این ساختار به LSTM اجازه می‌دهد تا اطلاعات مهم را در حافظه خود ذخیره کرده و اطلاعات غیرضروری را حذف کند. به این ترتیب، LSTM قادر است به طور موثر وابستگی‌های طولانی‌مدت در داده‌های ترتیبی را به خاطر بسپارد.



ما در این پروژه، با استفاده از شبکه‌های LSTM یک سیستم ترجمه پیاده‌سازی کردم. مراحلی که برای پیاده‌سازی این سیستم صورت گرفت، به شرح زیر است:

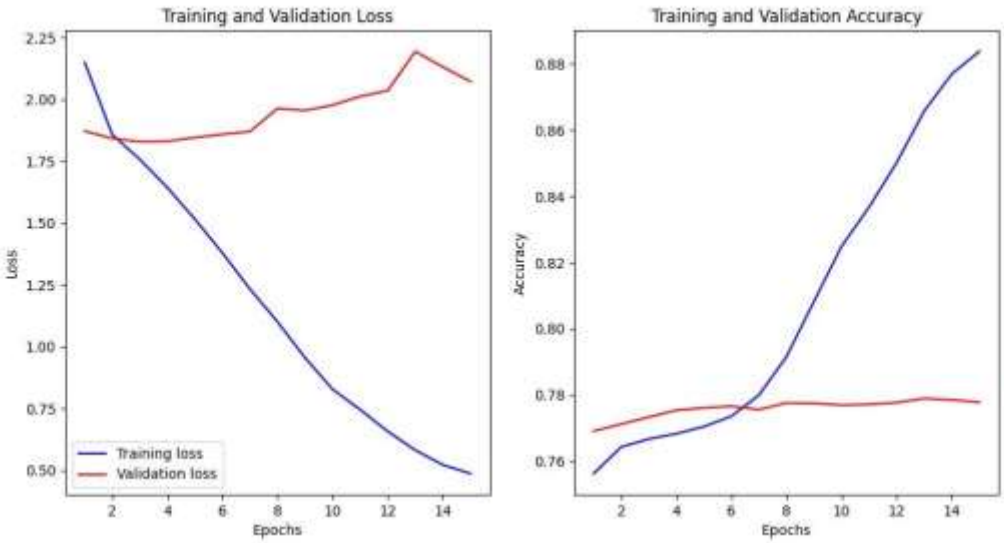
1. **بارگذاری داده‌ها:** داده‌های آموزشی و تست از دو فایل CSV حاوی جملات انگلیسی و ترجمه فارسی آن‌ها بارگذاری شدند. مجموعه آموزشی شامل جملات انگلیسی با ترجمه‌های فارسی است و مجموعه تست فقط جملات انگلیسی دارد.
2. **استخراج ویژگی‌ها با BERT:** از مدل BERT چندزبانه (Multilingual BERT) استفاده شد تا جملات انگلیسی به بردارهای ویژگی (یا تعبیه‌های متنی) تبدیل شوند. این بردارها شامل اطلاعات معنایی هستند و به عنوان ورودی به مدل LSTM داده می‌شوند. هر جمله انگلیسی پس از توکن‌سازی توسط BERT به برداری با طول ثابت 20 کلمه و ویژگی‌های معنایی تبدیل شد.
3. **پیش‌پردازش جملات فارسی:** جملات فارسی توکن‌سازی شده و به توالی‌های عددی تبدیل شدند که هر عدد نمایانگر یک کلمه در واژگان است. این توالی‌ها با استفاده از padding به طول ثابت تنظیم شدند تا ورودی‌های مدل دارای اندازه یکسانی باشند.
4. **ساخت مدل LSTM**: از یک شبکه LSTM با یک لایه LSTM استفاده شد. این لایه LSTM دارای تعدادی نورون است (۵۱۲) و تابع فعال‌سازی tanh برای پردازش داده‌ها استفاده شد. همچنین، برای جلوگیری از

بیش‌برازش (Overfitting)، لایه Dropout با مقدار ۰.۵ اضافه شدند. لایه نهایی مدل یک Dense layer با تابع فعال‌سازی Softmax است که توالی کلمات فارسی را به عنوان خروجی تولید می‌کند.

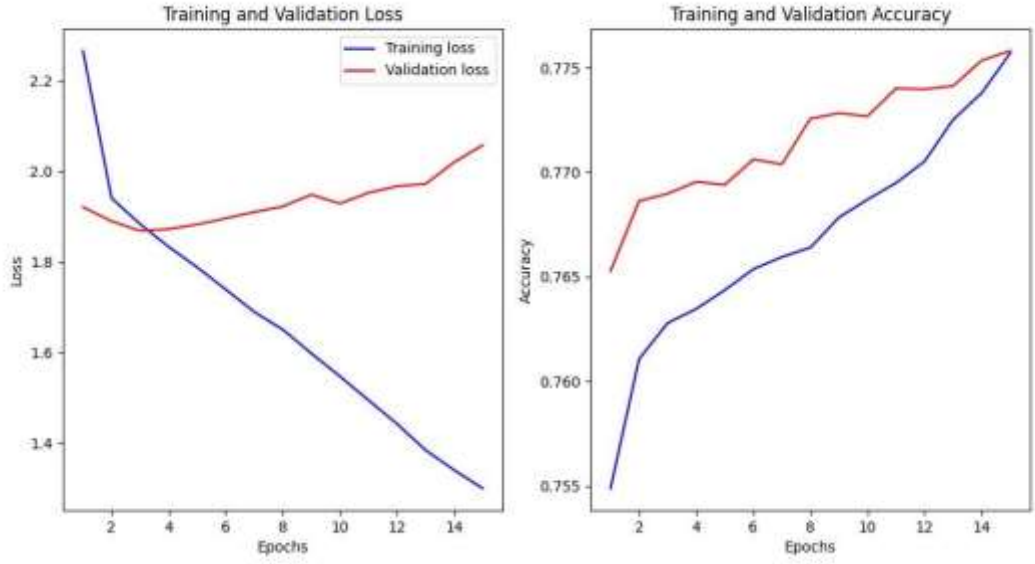
5. **آموزش و ارزیابی مدل:** مدل با استفاده از داده‌های آموزشی برای ۱۵ دوره (epoch) آموزش داده شد. داده‌های اعتبارسنجی (Validation) نیز برای ارزیابی عملکرد مدل در طول آموزش استفاده شد. تابع هزینه Sparse Categorical Crossentropy برای به حداقل رساندن خطای پیش‌بینی‌ها و بهینه‌ساز Adam برای بهبود سرعت یادگیری استفاده شدند.
6. **ترجمه و ارزیابی جملات تست:** پس از آموزش مدل، جملات انگلیسی در مجموعه تست به فارسی ترجمه شدند. فرآیند ترجمه شامل تبدیل جملات انگلیسی به بردارهای ویژگی با BERT و سپس استفاده از مدل LSTM برای تولید توالی ترجمه فارسی است. در نهایت، نتایج ترجمه نهایی در فایل CSV ذخیره شدند.

همانند مدل‌های مبتنی بر شبکه‌های RNN، در اینجا نیز مراحل مختلفی برای یافتن بهترین مدل مبتنی بر شبکه‌های LSTM سپری شد که در ادامه آن‌ها را شرح می‌دهیم.

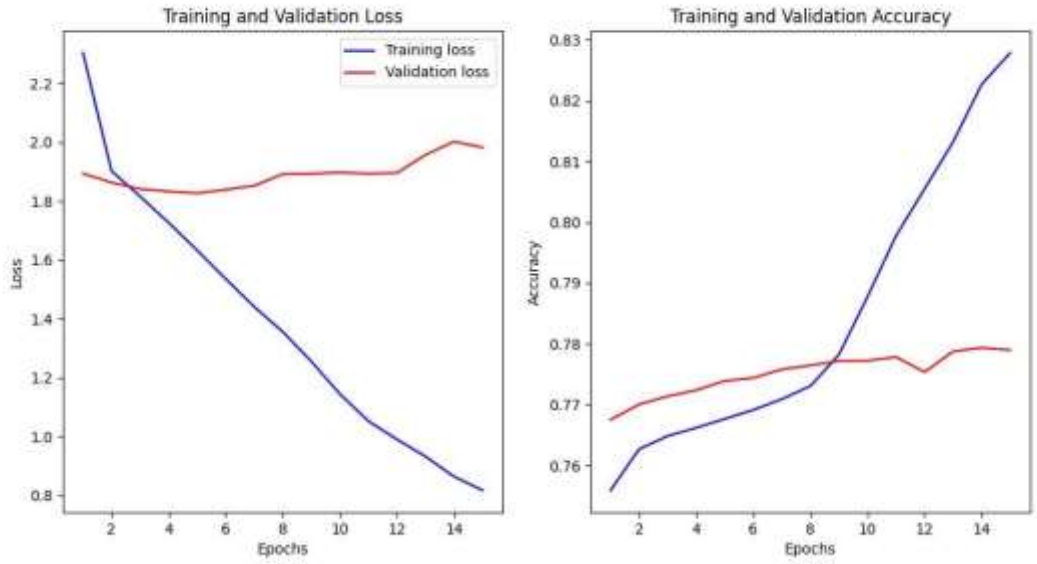
برای یافتن بهترین مدل برای پیاده سازی سیستم ترجمه با استفاده از LSTM، ابتدا مدلی طراحی کردیم که یک لایه LSTM دارای ۱۰۲۴ نورون بود. در این مدل مقدار Dropout را برابر ۰.۵ و تابع فعالسازی را تابع tanh قرار دادیم. با توجه به نتایج به دست آمده، در این مدل مقدار خطای ۲.۰۷ و مقدار دقت ۰.۷۷ حاصل شد. در ادامه، نمودار مدل اول قابل مشاهده است. همانطور که در نمودار نیز مشخص است، مقدار val_loss مدل زیاد و مقدار val_accuracy مدل کم است.



در مرحله بعد، مدل دوم را طراحی کردیم که شامل دولایه LSTM دو طرفه بود. در لایه اول ۵۱۲ نورون و در لایه دوم ۲۵۶ نورون قرار دادیم و تابع فعال‌سازی و همچنین Dropout را همانند مدل قبل قرار دادیم. در این مدل، دقت مدل نسبت به مدل قبل ثابت ماند، ولی توانستیم خطا را کمی کاهش دهیم و به مقدار ۲.۰۵ رساندیم. نمودار این مدل در ادامه آمده است.



در مدل سوم از یک لایه LSTM با ۵۱۲ نورون استفاده کردیم. در این مدل، همچنان، دقت مدل ثابت ماند اما مقدار خطا کاهش بیشتری داشت و به مقدار ۱.۹۸ رسید. در نهایت، این مدل را به عنوان بهترین مدل انتخاب کردیم. هرچند مدل LSTM نسبت به RNN برای ترجمه عملکرد بهتری داشت ولی باز هم خیلی قابل قبول نبود و مدل دچار overfitting شد. در پایان، نمودار مدل نهایی در ادامه قابل مشاهده است.



مدل‌های Sequence to Sequence:

مدل‌های Seq2Seq نوعی از مدل‌های یادگیری ماشین هستند که برای تبدیل یک دنباله از داده‌ها به دنباله‌ای دیگر طراحی شده‌اند و به طور گسترده در مسائل مختلفی مانند ترجمه ماشینی استفاده می‌شوند. این مدل‌ها معمولاً از دو بخش اصلی تشکیل می‌شوند: رمزگذار (Encoder) و رمزگشا (Decoder). رمزگذار وظیفه دارد که دنباله ورودی (مثلاً یک جمله به زبان انگلیسی) را به یک بردار فشرده تبدیل کند که شامل اطلاعات مهمی از ورودی است. برای این کار، اغلب از شبکه‌های عصبی بازگشتی مانند RNN، LSTM یا GRU استفاده می‌شود. در مرحله بعد، رمزگشا

این بردار را به دنباله خروجی (مثلاً جمله‌ای به زبان فارسی) تبدیل می‌کند. رمزگشا معمولاً مشابه رمزگذار عمل می‌کند، اما به صورت معکوس، یعنی یک بردار را به دنباله‌ای از کلمات تبدیل می‌کند.

در مراحل انجام عملیات، ابتدا دنباله ورودی به توکن‌ها تبدیل شده و توسط رمزگذار به یک بردار فشرده تبدیل می‌شود. سپس این بردار به رمزگشا منتقل شده و رمزگشا دنباله خروجی را به صورت گام به گام تولید می‌کند. به عنوان مثال، اگر جمله "How are you?" به مدل داده شود و قصد ترجمه آن به فارسی را داشته باشیم، رمزگذار جمله را به یک بردار فشرده (مثلاً یک بردار عددی ۷۶۸ بعدی) تبدیل کرده و رمزگشا جمله "چطوری؟" را به عنوان خروجی تولید می‌کند.

از زمان معرفی اولیه، مدل‌های Seq2Seq پیشرفت‌های زیادی کرده‌اند. در ابتدا از RNN و LSTM برای ساختار رمزگذار و رمزگشا استفاده می‌شد. با گذشت زمان، مکانیزم توجه (Attention Mechanism) به این مدل‌ها اضافه شد که امکان تمرکز بر بخش‌های مهم‌تر دنباله ورودی را در هر مرحله از ترجمه فراهم می‌کرد. سپس مدل‌های Transformer معرفی شدند که بدون استفاده از شبکه‌های بازگشتی، با مکانیزم توجه توانستند دنباله‌های طولانی‌تری از داده‌ها را سریع‌تر و دقیق‌تر ترجمه کنند. مدل‌های جدیدتری مانند GPT، BERT و T5 نیز از معماری Transformer بهره می‌برند و قابلیت‌های پیشرفته‌ای در ترجمه دارند.

ترجمه ماشینی یکی از کاربردهای اصلی این مدل‌ها است که جملات را از یک زبان به زبان دیگر تبدیل می‌کند. مدل‌های Seq2Seq به دلیل انعطاف‌پذیری و قدرت زیادشان، از ابزارهای مهم در ترجمه ماشینی و پردازش زبان طبیعی محسوب می‌شوند. در این پروژه، دو مدل Bart و T5 پیاده‌سازی شده‌اند که در ادامه هر کدام به طور کامل شرح داده شده‌اند.

مدل T5:

مدل T5 یا Text-To-Text Transfer Transformer یکی از مدل‌های قدرتمند مبتنی بر ترانسفورمر است که توسط تیم Google Research ارائه شده است. این مدل به گونه‌ای طراحی شده که تمامی وظایف پردازش زبان طبیعی (NLP) را به صورت یک مسئله تبدیل متن به متن (Text-to-Text) حل کند. ورودی و خروجی همه وظایف به صورت رشته‌ای از متن است و وظایف مختلف مانند ترجمه در قالب تولید متن قابل بیان هستند. T5 از معماری ترانسفورمر

استفاده می‌کند که یک ساختار Seq2Seq است. ویژگی کلیدی T5 این است که از مدل متن به متن برای تمامی وظایف پردازش زبان طبیعی بهره می‌برد. در این معماری، ورودی و خروجی هر دو به شکل متن هستند.

در فرآیند ترجمه، ورودی مدل جمله‌ای به زبان انگلیسی است که به یک نمایش برداری (embedding) تبدیل می‌شود. سپس، مدل وظیفه تولید جمله هدف به زبان فارسی را بر عهده دارد. T5 از مکانیزم self-attention برای یادگیری روابط بین کلمات در جمله ورودی استفاده می‌کند و همچنین از cross-attention برای برقراری رابطه بین ورودی رمزگذار و خروجی رمزگشا بهره می‌برد. برای هر وظیفه خاص مانند ترجمه، یک پرامپت متنی به مدل داده می‌شود، به عنوان مثال، جمله ورودی می‌تواند به شکل زیر باشد

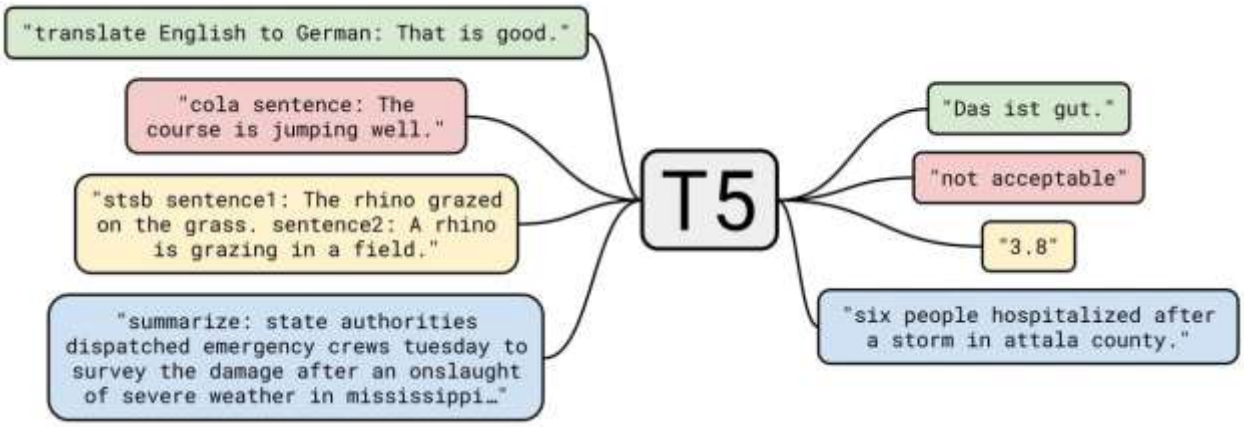
"Translate English to Persian: [input sentence]"

و مدل جمله هدف را در زبان مقصد تولید می‌کند.

T5 می‌تواند به طور مؤثر با داده‌های محدود به دلیل قدرت بالای یادگیری انتقالی (Transfer Learning) کار کند. پردازش متن به متن برای همگام‌سازی وظایف مختلف پردازش زبان بسیار مناسب است. یادگیری انتقالی فرایندی است که در آن یک مدل از پیش آموزش دیده‌شده که روی یک مجموعه داده یا وظیفه بزرگ آموزش دیده، برای وظیفه جدید مورد استفاده قرار می‌گیرد. هدف این است که از دانش و ویژگی‌هایی که مدل در مرحله آموزش اولیه یاد گرفته است، برای وظیفه دیگری استفاده شود. یادگیری انتقالی معمولاً شامل دو مرحله است: نخست، مدل روی یک مجموعه داده بزرگ و عمومی آموزش داده می‌شود تا ویژگی‌های کلی را یاد بگیرد و سپس برای یک وظیفه خاص با یک مجموعه داده جدید و کوچکتر تنظیم دقیق می‌شود. همچنین، Downstream learning به مرحله‌ای اشاره دارد که پس از یادگیری انتقالی انجام می‌شود. وظایف پایین‌دستی، همان وظایف خاصی هستند که مدل باید پس از پیش‌تمرین روی آن‌ها بهینه‌سازی شود. یادگیری انتقالی شامل استفاده از مدل از پیش آموزش‌دیده‌شده و تنظیم آن برای وظایف جدید است، در حالی که یادگیری پایین‌دستی به وظایف خاصی اشاره دارد که پس از پیش‌تمرین مدل انجام می‌شود. به عبارت دیگر، یادگیری پایین‌دستی زیرمجموعه‌ای از یادگیری انتقالی است و بر روی تنظیم و بهینه‌سازی مدل برای یک وظیفه خاص تمرکز دارد.

ما در این پروژه، ابتدا مدل T5 از پیش آموزش دیده شده را از سایت Huggingface بر روی داده‌های خود تست کردیم. خروجی این مدل به طور رسمی جملات را ترجمه می‌کرد که مناسب ترجمه زیرنویس فیلم نیست. به همین دلیل، ما این مدل را با داده‌های خودمان و در ۴ دوره (epoch) آموزش دادیم. در نهایت، با باز تنظیم این مدل و

آموزش داده‌های خود، به نتایج بهتری رسیدیم و خروجی غیر رسمی مد نظر برای زیرنویس‌های یک فیلم را دریافت کردیم.

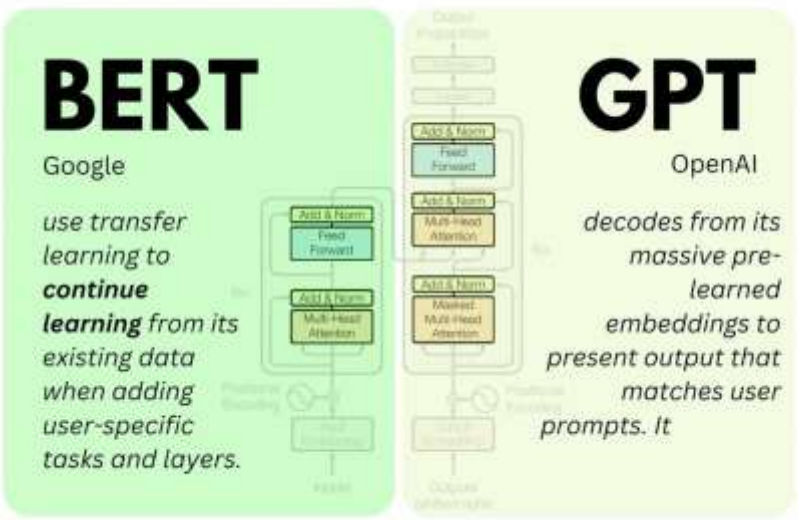


مدل BART:

این مدل یکی از مدل‌های پیشرفته مبتنی بر معماری ترانسفورمر است که توسط تیم Google Research در سال ۲۰۱۹ ارائه شد. این مدل از یک رمزگذار دوطرفه (مشابه BERT) و یک رمزگشای چپ به راست (مشابه GPT) بهره می‌برد که در تسک‌های مختلف پردازش زبان طبیعی، از جمله ترجمه، بسیار مؤثر عمل می‌کند.

مدل BERT یا Bidirectional Encoder Representations from Transformers یک مدل قدرتمند است که تمرکزش بر درک متن است. رمزگذار BERT به صورت دوطرفه عمل می‌کند، به این معنا که می‌تواند متن را از هر دو جهت (چپ و راست) بررسی کند و روابط معنایی بین کلمات را به‌خوبی درک کند. BERT برای تسک‌هایی مانند طبقه‌بندی و استخراج اطلاعات بسیار موفق بوده و به دلیل این توانایی، در درک متن‌های پیچیده کمک شایانی می‌کند.

از سوی دیگر، مدل GPT یا Generative Pretrained Transformer بر تولید متن تمرکز دارد. برخلاف BERT که رمزگذاری دوطرفه دارد، GPT یک مدل یک‌طرفه است که متن را از چپ به راست تولید می‌کند. این مدل برای تسک‌هایی مانند تکمیل متن یا تولید جملات جدید کاربرد دارد و در محیط‌هایی که تولید زبان به شکل خودبازگشتی انجام می‌شود، عملکرد خوبی دارد.



مدل BART با ترکیب ویژگی‌های رمزگذار دوطرفه BERT و رمزگشای یک‌طرفه GPT، توانسته است مدلی چندمنظوره ایجاد کند که در ترجمه، خلاصه‌سازی و سایر تسک‌های تولید متن مؤثر است. در تسک ترجمه، ورودی مدل به صورت یک جمله در زبان مبدا پردازش شده و سپس توسط رمزگشا به جمله‌ای در زبان مقصد تبدیل می‌شود. به دلیل استفاده از مکانیزم‌های یادگیری انتقالی ، BART حتی با داده‌های آموزشی محدود، عملکرد خوبی دارد.

BART در مقایسه با مدل‌های مشابه، به‌ویژه در ترجمه ماشین و تولید جملات روان و معنادار، نتایج قابل‌توجهی ارائه کرده و از ترکیب قدرت‌های BERT و GPT برای دستیابی به ترجمه‌های دقیق‌تر بهره می‌برد.

ما در این پروژه، ابتدا با رمزگذار BERT و رمزگشا GPT مدل BART را تا حدود ۳۰ دوره (epoch) بر روی دیتای جمع‌آوری شده‌ی خود آموزش دادیم. با این حال، نتایج خوبی در داده‌های تست مشاهده نشد. بنابراین، در گام بعدی، از مدل پیش‌آموزش دیده شده در سایت Huggingface استفاده کردیم. در این مدل، داده‌های جمع‌آوری شده توسط خودمان را تا ۳ دوره (epoch) با استفاده از این مدل آموزش دادیم و نتایج به نسبت بهتری به دست آمد.

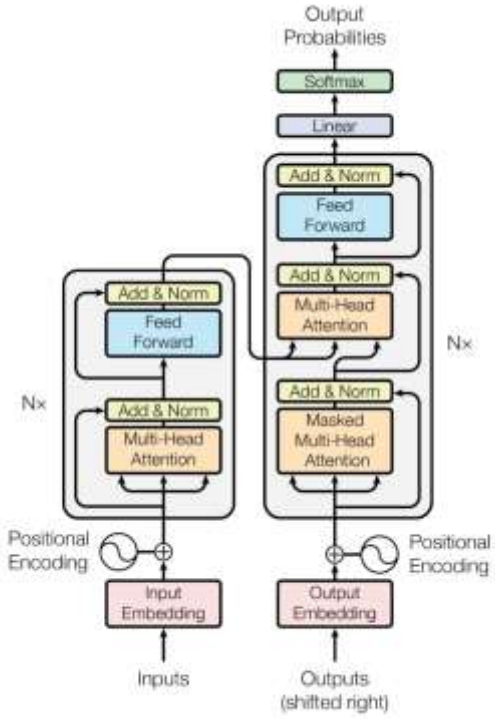


Figure 1: The Transformer - model architecture.

گام سوم - سیستم مترجم مبتنی بر مدل‌های LLM:

یک مدل زبانی عظیم (Large Language Model) یا به اختصار LLM با استفاده از معماری شبکه‌های عصبی، بر روی حجم بسیار گسترده‌ای از داده‌ها مانند متن‌های کتاب‌ها، مقالات و مکالمات (در مقیاس پتابایت) آموزش می‌بینند. این مدل‌ها قادر به درک رابطه‌های پنهان بین توکن‌های مختلف یک متن هستند و می‌توانند با دریافت یک پرامپت (ورودی اولیه)، توکن بعدی را پیش‌بینی کرده و متن را به طور منطقی ادامه دهند. مقیاس بزرگی که این مدل‌ها بر اساس آن آموزش می‌بینند، باعث می‌شود بتوانند متون منسجم و متنوعی تولید کنند و با ارائه داده‌های بیشتر، عملکرد خود را بهبود دهند.

در این مرحله، یک سیستم ترجمه مبتنی بر مدل‌های LLM پیاده‌سازی شد. برای این منظور، با استفاده از بخش Wrapper در سایت متیس و کلید API آن، از کتابخانه Langchain در پایتون بهره گرفته شد تا فرآیند ترجمه انجام گیرد. جزئیات مراحل به شرح زیر است:

- درخواست به سایت متیس با استفاده از کلید API و اتصال به مدل GPT-4o.
- استفاده از کلاس LLM در LangChain و بهره‌گیری از این مدل در آن.
- تعریف دستور با کمک کتابخانه‌ی PromptTemplate در LangChain برای ترجمه‌ی غیررسمی متون، متناسب با زیرنویس فیلم‌ها.
- دریافت خروجی از طریق توابع موجود در LangChain.

در این مرحله ابتدا از مدل GPT-3.5-Turbo-0125 که آخرین نسخه از GPT-3.5-Turbo است، استفاده شد. خروجی این مدل بر روی داده‌های تست با استفاده از امتیاز BLEU ارزیابی شد و مقدار ۰.۰۴۷ به دست آمد. پس از این، از مدل GPT-4o استفاده شد که نسخه‌ای به‌روزرسانی‌شده از GPT-4 است و تغییراتی را شامل می‌شود که ممکن است کیفیت یا سرعت ترجمه را بهبود بخشد. این نسخه برای بهینه‌سازی‌های جدیدتر طراحی شده و می‌تواند بهبودهایی مانند درک بهتر زمینه یا پردازش سریع‌تر را ارائه دهد. در نهایت، خروجی این مدل هم از نظر متریک و هم از نظر بررسی چشمی موثرتر بود.

گام چهارم - ارزیابی مدل‌های پیاده‌سازی شده:

برای ارزیابی مدل‌های مختلفی که در این پروژه انجام شد، از امتیاز BLEU استفاده شده است. معیار BLEU (Bilingual Evaluation Understudy) یک شاخص خودکار برای ارزیابی کیفیت ترجمه‌های ماشینی است. به‌جای ارزیابی‌های ذهنی انسانی، این معیار بر اساس مقایسه n-gramهای مشترک بین ترجمه ماشین و ترجمه‌های مرجع انسانی عمل می‌کند. BLEU به دلیل سرعت و دقت در ارزیابی خودکار عملکرد سیستم‌های ترجمه، به یکی از معیارهای استاندارد در این حوزه تبدیل شده است.

نحوه محاسبه این معیار به شرح زیر است:

1. **n-gramها:** BLEU کلمات یا گروه‌هایی از کلمات (مانند دوتایی‌ها، سه‌تایی‌ها و غیره) را بین ترجمه ماشینی و ترجمه مرجع مقایسه می‌کند.

2. **دقت (Precision):** بررسی می‌کند که چه تعداد از این n-gramها در ترجمه ماشینی با ترجمه مرجع مطابقت دارند.
3. **جریمه برای ترجمه‌های کوتاه (Brevity Penalty):** اگر ترجمه ماشینی به‌طور قابل‌توجهی کوتاه‌تر از ترجمه مرجع باشد، جریمه‌ای اعمال می‌شود تا از کاهش کیفیت جلوگیری شود.
4. **امتیاز BLEU:** در نهایت، امتیازی بین 0 تا 1 محاسبه می‌شود که نشان‌دهنده کیفیت ترجمه است(1 به معنای بهترین ترجمه ممکن است).

ما این امتیاز را بر روی هر کدام از مدل‌هایی که پیاده‌سازی کردیم، با استفاده از داده‌های تستی که در ابتدا ایجاد کرده بودیم، محاسبه کردیم. در جدول زیر، امتیازات به دست آمده برای هر کدام از مدل‌ها آورده شده است:

Model Name	BLEU Score
RNN	0.006
LSTM	0.030
Bart	0.076
T5 (Original)	0.100
T5 (Fine Tune)	0.130
LLM (GPT-3.5-turbo-0125)	0.047
LLM (GPT-4o)	0.082

گام آخر – داکرایز کردن پروژه:

در این بخش، مدل T5-fine tuned با استفاده از کد API داکرایز شد تا امکان اجرا در محیط‌های مختلف و بهبود مقیاس‌پذیری و مدیریت فراهم شود.

مراحل انجام کار به شرح زیر است:

1. **آماده‌سازی و ذخیره مدل:** مدل را آموزش داده و ذخیره کردیم.
2. **طراحی API با Flask:** یک API ساده با استفاده از Flask طراحی شد که متن انگلیسی را دریافت کرده و آن را به فارسی ترجمه می‌کند. از پروتکل HTTP و روش POST برای ارسال و دریافت داده‌ها استفاده کردیم.
3. **داکرایز کردن اپلیکیشن:** یک فایل داکر ایجاد شد که شامل مراحل نصب پکیج‌ها و تنظیم محیط برای اجرای Flask است.
4. **ساخت و راه‌اندازی کانتینر داکر:** پس از ایجاد فایل داکر، ایمیج ساخته و راه‌اندازی شد.
5. **استفاده از Docker Compose:** برای مدیریت بهتر و مقیاس‌پذیری، از Docker Compose استفاده شد که مزایایی مانند سازماندهی بهتر و سازگاری با کانتینرهای مختلف دارد.
6. **تست API:** با استفاده از Postman یا Insomnia، API مورد آزمایش قرار گرفت و ترجمه فارسی متن انگلیسی از طریق پورت محلی دریافت شد.

داکرایز کردن API، راهکاری موثر برای مدیریت و مقیاس‌پذیری مدل ترجمه فراهم کرد. این روش امکان استفاده از مدل در پروژه‌های مختلف را فراهم کرده و به ما اجازه می‌دهد مدل را در طول زمان بهبود بخشیم.

نتیجه‌گیری:

در این پروژه، سیستم ترجمه ماشینی برای ترجمه زیرنویس‌های انگلیسی به فارسی طراحی و ارزیابی شد. با استفاده از مدل‌های مختلف از جمله RNN، LSTM، مدل‌های مبتنی بر ترانسفورمر مانند BART و T5، و همچنین مدل‌های

بزرگ زبان (LLM) نظیر GPT-3.5 و GPT-4، نتایج ترجمه مورد بررسی قرار گرفت. داده‌های متنوعی شامل مجموعه‌های موازی انگلیسی-فارسی از منابع مختلف جمع‌آوری و پردازش شدند تا مدل‌ها به خوبی آموزش ببینند.

مدل‌های کلاسیک‌تر مانند RNN و LSTM با چالش‌هایی از جمله دقت پایین و محدودیت در پردازش جملات بلند مواجه بودند. از سوی دیگر، مدل‌های مبتنی بر ترانسفورمر، به‌ویژه T5 و BART، عملکرد بهتری داشتند و توانستند ترجمه‌های روان‌تر و دقیق‌تری ارائه دهند. همچنین، با استفاده از مدل‌های بزرگ زبان مانند GPT-4، دقت ترجمه به طور قابل توجهی بهبود یافت. ارزیابی نهایی با استفاده از معیار BLEU نشان داد که مدل‌های پیشرفته‌تر به‌ویژه پس از تنظیم دقیق (fine-tuning)، قادر به تولید ترجمه‌های باکیفیت‌تری هستند. به‌طور کلی، این پروژه نشان داد که استفاده از مدل‌های پیشرفته‌تر و تکنیک‌های بهینه‌سازی داده، دقت و کارایی ترجمه‌ها را بهبود می‌بخشد.

پیشنهادات برای کارهای آینده:

در مدل‌هایی که به کمک شبکه‌های RNN و LSTM پیاده‌سازی کردیم، نتایج خوبی حاصل نشد. برای جلوگیری از overfitting در مدل و هچنین کسب نتایج بهتر، چندین راهکار وجود دارد که می‌توان از آنها برای بهبود مدل در آینده استفاده کرد:

1. **استفاده از Early stopping:** این روش، می‌تواند از ادامه آموزش مدل در زمانی که عملکرد مدل در داده‌های اعتبارسنجی بهبود پیدا نمی‌کند یا بدتر می‌شود، جلوگیری کند.
2. **افزودن L2 Regularization:** این کار به مدل کمک می‌کند تا از پیچیدگی بیش از حد و overfitting جلوگیری کند.
3. **افزایش تعداد داده ها:** می‌توان تعداد داده ها را افزایش داد تا از overfitting جلوگیری کرد.
4. **افزایش نرخ Dropout:** افزایش میزان dropout به مدل کمک می‌کند تا وابستگی به نودهای خاص در هر لایه را کاهش دهد.

همچنین، به طور کلی نیز می‌توان در آینده موارد زیر را در نظر گرفت:

- **گسترش و بهبود مجموعه داده‌ها:** هرچند مجموعه داده‌های فعلی حاوی جملات موازی انگلیسی و فارسی است، اما افزایش حجم داده‌ها و استفاده از داده‌های بیشتر و متنوع‌تر می‌تواند به بهبود دقت مدل‌ها کمک کند. به‌ویژه، گسترش داده‌ها به موضوعات غیررسمی و محاوره‌ای می‌تواند کاربرد مدل‌ها را در شرایط مختلف افزایش دهد.
- **تنظیم دقیق‌تر مدل‌های LLM:** با توجه به عملکرد موفقیت‌آمیز مدل‌های بزرگ زبان (LLM)، پیشنهاد می‌شود که مدل‌های جدیدتر مانند GPT-4 با تنظیم دقیق‌تر بر روی مجموعه داده‌های خاص فارسی ارزیابی شوند. همچنین، استفاده از نسخه‌های بزرگ‌تر و قوی‌تر این مدل‌ها می‌تواند به بهبود دقت ترجمه‌ها کمک کند.
- **استفاده از تکنیک‌های پیشرفته بهینه‌سازی:** به‌کارگیری تکنیک‌هایی مانند Early Stopping و L2 Regularization در فرآیند آموزش می‌تواند از مشکل بیش‌برازش (Overfitting) جلوگیری کند و کارایی مدل‌ها را بهبود بخشد. این تکنیک‌ها به‌ویژه در مدل‌هایی که با داده‌های بزرگ و پیچیده سروکار دارند، بسیار موثر هستند.
- **بررسی و استفاده از معیارهای ارزیابی مختلف:** ارزیابی ترجمه‌ها با معیار BLEU اطلاعات مفیدی فراهم کرد، اما برای داشتن دید کامل‌تری از کیفیت ترجمه‌ها، می‌توان از معیارهای دیگری نظیر METEOR، ROUGE و TER استفاده کرد. این معیارها جنبه‌های مختلفی از کیفیت ترجمه را پوشش می‌دهند و می‌توانند به ارزیابی دقیق‌تر کمک کنند.
- **بهبود عملکرد در جملات طولانی:** همانطور که در این پروژه مشخص شد، مدل‌های RNN و LSTM با مشکل نگهداشتن اطلاعات در جملات طولانی مواجه هستند. استفاده از مدل‌های دقت بالاتر مانند ترانسفورمر و به‌کارگیری تکنیک‌هایی مانند توجه (Attention Mechanism) برای مدیریت بهتر روابط بین کلمات در جملات طولانی پیشنهاد می‌شود.
- **ترجمه بهینه‌تر برای فیلم‌ها و محتوای چندرسانه‌ای:** هدف اصلی این پروژه ترجمه زیرنویس فیلم‌ها بود. به منظور بهبود بیشتر، می‌توان بر روی تطبیق مدل‌ها با نیازهای خاص محتوای چندرسانه‌ای تمرکز کرد، مانند حفظ همزمانی ترجمه با گفتار، یا مدیریت ترجمه‌های محاوره‌ای و اصطلاحات خاص.