به نام خدا

# کارگاه آشنایی با ابزار های هضم و حرف درس مبانی پردازش زبان و گفتار دکتر مرضیه داودآبادی فراهانی

# فرزان رحمانی 99521271

## سوال اول

ایجاد یک محصول تشخیص گفتار خودکار (ASR) شامل یک فرآیند توسعه چندوجهی است که رشته‌های مختلفی از جمله پردازش سیگنال، یادگیری ماشین و مهندسی نرم‌افزار را ادغام می‌کند. در اینجا یک مرور کلی از مراحل مربوطه آورده شده است:

1. تحقیقات بازار و تجزیه و تحلیل نیاز (Market Research and Requirement Analysis)

هدف: شناخت بازار هدف و تعریف الزامات محصول.

* موارد استفاده را شناسایی کنید: برنامه‌های خاص ASR را تعیین کنید (مانند transcription services, virtual assistants, voice-activated controls).
* تجزیه و تحلیل رقبا: محصولات ASR موجود را مطالعه کنید تا شکاف ها و فرصت ها را شناسایی کنید.
* نیازهای کاربر را جمع آوری کنید: نظرسنجی، مصاحبه و گروه های متمرکز را برای جمع آوری اطلاعات از کاربران بالقوه انجام دهید.

2. مطالعه امکان سنجی و برنامه ریزی (Feasibility Study and Planning)

هدف: ارزیابی امکان سنجی فنی و برنامه ریزی فرآیند توسعه.

* امکان سنجی فنی: در دسترس بودن و بلوغ فناوری های لازم (مانند الگوریتم های تشخیص گفتار، الزامات سخت افزاری) را ارزیابی کنید.
* برنامه ریزی پروژه: یک برنامه دقیق پروژه شامل جدول زمانی، منابع و بودجه ایجاد کنید.

3. جمع آوری و پیش پردازش داده ها (Data Collection and Preprocessing)

هدف: جمع آوری و آماده سازی داده ها برای آموزش مدل های ASR.

* جمع آوری داده ها: مجموعه داده بزرگی از فایل های ضبط شده صوتی را با رونویسی متن مربوطه به دست آورید. این مجموعه داده باید متنوع(مثلا اخبار، فیلم ها و ...) باشد و لهجه‌ها، گویش‌ها و شرایط گفتاری مختلف را پوشش دهد.
* حاشیه نویسی داده ها(Data Annotation): اطمینان حاصل کنید که داده های صوتی به طور دقیق رونویسی و برچسب گذاری شده اند.
* داده افزایی(Data Augmentation): با افزودن نویز، تغییر زیر و بم یا شبیه سازی محیط های مختلف ضبط، تنوع مجموعه داده را افزایش دهید.
* پیش پردازش: فایل های صوتی را به فرمت مناسب تبدیل کنید، سطوح صدا را عادی کنید و ویژگی ها را استخراج کنید (به عنوان مثال، Mel-Frequency Cepstral Coefficients - MFCCs).

4. توسعه مدل (Model Development)

هدف: توسعه و آموزش مدل ASR.

* معماری مدل را انتخاب کنید: یک معماری مدل مناسب مانند مدل‌های مارکوف پنهان (HMM)، شبکه‌های عصبی عمیق (DNN)، شبکه‌های عصبی بازگشتی (RNN) یا مدل‌های مبتنی بر ترانسفورماتور را انتخاب کنید.
* آموزش مدل: مدل را با استفاده از داده های از پیش پردازش شده آموزش دهید. این شامل تنظیم مکرر پارامترهای مدل برای به حداقل رساندن خطاهای تشخیص است.
* اعتبار سنجی و آزمایش: برای ارزیابی عملکرد مدل و جلوگیری از برازش بیش از حد، داده ها را به مجموعه های آموزشی، اعتبار سنجی و آزمایش تقسیم کنید.

5. یکپارچه سازی و بهینه سازی (Integration and Optimization)

هدف: ادغام مدل ASR در یک محصول قابل استقرار و بهینه سازی عملکرد آن.

* یکپارچه‌سازی سیستم: مدل ASR را در پلتفرم نرم‌افزاری یا سخت‌افزاری که در آن مورد استفاده قرار می‌گیرد قرار دهید (به عنوان مثال، برنامه تلفن همراه، سرویس ابری).
* بهینه سازی تأخیر: اطمینان حاصل کنید که سیستم ASR به اندازه کافی سریع برای برنامه های کاربردی بلادرنگ پاسخ می دهد.
* بهینه سازی دقت: مدل را برای بهبود دقت برای موارد استفاده یا محیط های خاص تنظیم کنید.

6. طراحی رابط کاربری و تجربه (User Interface and Experience Design)

هدف: طراحی یک رابط کاربری و تجربه کاربر پسند.

* طراحی رابط: یک رابط کاربری (UI) ایجاد کنید که به کاربران اجازه می دهد به طور موثر با سیستم ASR تعامل داشته باشند.
* تست تجربه کاربر (UX): تست های قابلیت استفاده را برای جمع آوری بازخورد و اصلاح UI/UX انجام دهید.

7. تست و تضمین کیفیت (Testing and Quality Assurance)

هدف: اطمینان حاصل کنید که محصول ASR قوی و قابل اعتماد است.

* آزمایش بتا: یک نسخه بتا را برای گروهی از کاربران منتخب منتشر کنید تا هر مشکلی را شناسایی کنید.
* ارزیابی عملکرد: سیستم را تحت شرایط مختلف آزمایش کنید تا از عملکرد ثابت اطمینان حاصل کنید.
* مدیریت خطا: مکانیسم هایی را برای رسیدگی به خطاهای تشخیص به خوبی اجرا کنید.

8. استقرار و نگهداری (Deployment and Maintenance)

هدف: استقرار محصول ASR و ارائه پشتیبانی مداوم.

* استقرار: محصول را از طریق کانال های مناسب (مانند فروشگاه های اپلیکیشن، پلتفرم های ابری) به بازار عرضه کنید.
* نظارت: به طور مداوم بر عملکرد سیستم و بازخورد کاربران نظارت کنید.
* به روز رسانی و نگهداری: به طور منظم سیستم را برای رفع اشکالات، بهبود عملکرد و افزودن ویژگی های جدید به روز کنید.

9. بهبود مستمر (Continuous Improvement)

هدف: بهبود مستمر محصول ASR بر اساس بازخورد کاربر و پیشرفت های تکنولوژیکی.

* بازخورد کاربر: جمع آوری و تجزیه و تحلیل بازخورد برای شناسایی زمینه های بهبود.
* به روز رسانی های تکنولوژیکی: جدیدترین پیشرفت ها در فناوری تشخیص گفتار را در خود جای دهید.
* گسترش: قابلیت های محصول را گسترش دهید (به عنوان مثال، اضافه کردن پشتیبانی برای زبان های بیشتر یا واژگان تخصصی).

ابزارها و فناوری های پیشنهادی

* زبان های برنامه نویسی: پایتون، سی پلاس پلاس، جاوا
* چارچوب ها و کتابخانه ها: TensorFlow، PyTorch، Kaldi، DeepSpeech
* خدمات ابری: AWS، Google Cloud، Azure برای استقرار مقیاس پذیر و قدرت محاسباتی
* APIها: APIهای تشخیص گفتار مانند Google Speech-to-Text، Amazon Transcript یا راه حل های سفارشی ساخته شده یا تولید راه حل ها توسط خودتان

نتیجه

توسعه یک محصول ASR یک فرآیند پیچیده و مکرر است که نیاز به همکاری در چندین رشته دارد. با دنبال کردن این مراحل و استفاده از ابزارها و فناوری های مدرن، می توانید یک راه حل موثر و رقابتی ASR ایجاد کنید.

## سوال دوم

هنگامی که چندین مدل ASR دارید که فهرستی از رونویسی‌های احتمالی را ارائه می‌کنند، انتخاب 16 خروجی نهایی شامل جمع‌آوری و رتبه‌بندی مؤثر این احتمالات است. در اینجا دو روش عملی برای رسیدن به این هدف آمده است:

1. روش گروهی با رای گیری و رتبه بندی (Ensemble Method with Voting and Ranking)

این روش با در نظر گرفتن پیش‌بینی هر مدل به عنوان یک رأی، خروجی‌های مدل‌های مختلف را جمع‌آوری می‌کند. لیست خروجی نهایی بر اساس فراوانی و رتبه بندی این آرا تعیین می شود.

مراحل:

1. جمع آوری پیش بینی ها: 16 پیش بینی برتر از هر مدل را جمع آوری کنید.
2. مکانیسم رأی گیری: تعداد وقوع هر پیش بینی منحصر به فرد در همه مدل ها را بشمارید. رتبه هر پیش‌بینی در فهرست خروجی مدل را می‌توان وزن کرد تا به پیش‌بینی‌هایی که توسط مدل‌ها رتبه‌بندی شده‌اند، اولویت بیشتری داده شود.
3. محاسبه امتیاز: به پیش بینی ها بر اساس رتبه و فراوانی آنها در بین مدل ها امتیاز بدهید. به عنوان مثال، پیش‌بینی‌هایی که رتبه اول را توسط یک مدل کسب می‌کنند، امتیاز بالاتری نسبت به پیش‌بینی‌هایی که رتبه‌های پایین‌تر دارند، می‌گیرند.
4. پیش بینی های رتبه بندی: نمرات هر پیش بینی منحصر به فرد را جمع کنید.
5. انتخاب نهایی: 16 پیش بینی برتر با بالاترین امتیاز را انتخاب کنید.

مثال:

* Model A’s top 16 predictions: [A1, A2, A3, ..., A16]
* Model B’s top 16 predictions: [B1, B2, B3, ..., B16]
* Model C’s top 16 predictions: [C1, C2, C3, ..., C16]

اگر A1، B1 و C1 یکسان باشد، به دلیل حضور مکرر و با رتبه بالا، امتیاز ترکیبی بالاتری می گیرد. بر اساس این نمرات جمع و رتبه بندی کنید تا به 16 نهایی برسید.

2. روش احتمال میانگین وزنی (Weighted Average Probability Method)

این روش شامل استفاده از امتیازات احتمالی است که به هر پیش‌بینی توسط مدل‌های مختلف اختصاص داده می‌شود و این امتیازها را برای رتبه‌بندی پیش‌بینی‌ها ترکیب می‌کند.

مراحل:

1. جمع آوری پیش بینی ها و احتمالات: برای هر مدل، 16 پیش بینی برتر را به همراه امتیازات احتمالی آنها جمع آوری کنید.
2. Normalize Probabilities: امتیازات احتمالی هر مدل را در صورتی که قبلاً در همان مقیاس نیستند، عادی کنید.
3. جمع وزنی احتمالات: برای هر پیش‌بینی منحصربه‌فرد، مجموع وزنی امتیازات احتمال آن را از مدل‌های مختلف محاسبه کنید. وزن مدل‌های مختلف را بر اساس عملکرد یا قابلیت اطمینان قبلی آنها تعیین کنید.
4. مجموع احتمالات: مجموع احتمالات وزنی برای هر پیش بینی.
5. رتبه بندی نهایی: پیش بینی ها را بر اساس احتمالات وزنی جمع آوری شده رتبه بندی کنید.
6. انتخاب نهایی: 16 پیش‌بینی برتر با بیشترین احتمالات را انتخاب کنید.

مثال:

* پیش بینی A1 مدل A با احتمال 0.9
* پیش بینی A1 مدل B با احتمال 0.85
* پیش بینی A1 مدل C با احتمال 0.8

اگر وزن مدل A، 0.4 باشد، مدل B، 0.3 و مدل C، 0.3 است:

احتمال وزنی برای A1 = (0.9 \* 0.4) + (0.85 \* 0.3) + (0.8 \* 0.3) = 0.36 + 0.255 + 0.24 = 0.855

برای همه پیش بینی های منحصر به فرد تکرار کنید و 16 مورد برتر را بر اساس این امتیازات جمع آوری شده انتخاب کنید.

\*\*\* می‌توانستیم از روش های ساده تری مانند روش های زیر استفاده کنیم ولی دو روش بالا robust تر هستند:

* انتخاب 16 خروجی ای که بیشترین احتمال را دارند مستقل از مدل. (اگر تکراری داشتیم سراغ احتمال بعدی برویم)
* برداشتن top-k احتمال از هر مدل برای مثال اگر 8 مدل داریم top-2 بیشترین احتمال را برداریم. (اگر تکراری داشتیم سراغ احتمال بعدی برویم)

ترکیب روش ها برای نتایج قوی (Combining Methods for Robust Results)

در عمل، ترکیب این روش ها می تواند نتایج قوی تری به همراه داشته باشد. برای مثال، می‌توانید از مکانیسم رأی‌گیری گروهی برای فیلتر کردن و رتبه‌بندی متداول‌ترین پیش‌بینی‌ها استفاده کنید و سپس از روش احتمال میانگین وزنی برای تنظیم دقیق رتبه‌بندی نهایی بر اساس امتیازات احتمالی استفاده کنید.

نتیجه

با استفاده از روش‌های گروهی مانند رای‌گیری و رتبه‌بندی یا احتمالات میانگین وزنی، می‌توانید به طور موثر نقاط قوت مدل‌های ASR متعدد را برای ایجاد فهرست نهایی رونویسی‌های(transcriptions) با کیفیت بالا ترکیب کنید. این روش‌ها به استفاده از قابلیت‌های متنوع مدل‌های مختلف برای بهبود دقت کلی و قابلیت اطمینان خروجی‌های نهایی کمک می‌کنند.

## سوال سوم

در حوزه پردازش متن و پردازش زبان طبیعی (NLP)، هر یک از این توابع نقش مهمی در تجزیه و تحلیل، تبدیل و درک متن دارند. در اینجا توضیح مفصلی در مورد هر تابع آورده شده است:

1. POS Tagger (تحلیل صرفی)

تحلیل صرفی (POS Tagger): یک برچسب POS به هر کلمه در یک جمله تگ های بخشی از گفتار اختصاص می دهد که نشان می دهد آیا یک کلمه یک اسم، فعل، صفت، قید و غیره است یا خیر. این به درک ساختار دستوری کمک می کند. از جمله

* مثال: " در می‌زند و دوباره وارد می‌شود. همه می‌گویند قدمش پربرکت است."
* برچسب‌ها:
  + در NOUN / می‌زند VERB / و CCONJ / دوباره ADV / وارد NOUN / می‌شود VERB / . PUNCT
  + همه PRON / می‌گویند VERB / قدمش NOUN / پربرکت ADJ / است VERB / . PUNCT

2. Chunker (تجزیه سطحی)

Chunker: همچنین به عنوان تجزیه سطحی (shallow parsing) شناخته می‌شود، تیکه‌سازی(chunking) شامل تقسیم‌بندی و برچسب‌گذاری توالی‌های چند نشانه‌ای (تکه‌ها) در یک جمله، مانند عبارات اسمی (NP)، عبارات فعل (VP) و غیره است. ساختار نحوی سطح بالاتری نسبت به POS ارائه می‌کند. برچسب زدن

* مثال: "همه می‌گویند قدمش پربرکت است."
  + تکه ها: [همه NP] [می‌گویند VP] [قدمش NP] [پر برکت ADJP] [است VP] .

3. Stemmer

Stemmer: یک stemmer کلمات را با حذف پسوندها به شکل پایه یا ریشه آنها کاهش می دهد. این به ساده‌سازی متن برای کارهای مختلف NLP کمک می‌کند، اما همیشه کلمات معتبری تولید نمی‌کند.

* مثال:
  + "running", "runs", "runner" → "run"
  + "کتاب" → "کتاب‌ها"

4. Lematizer

Lemmatizer: یک lemmatizer همچنین کلمات را به شکل پایه یا ریشه آنها کاهش می دهد که به عنوان لم شناخته می شود، اما بافت و بخشی از گفتار را برای تولید کلمات معتبر در نظر می گیرد. مثلا برای پیدا کردن ریشه افعال فارسی(بن ماضی، بن مضارع) می‌توانیم از آن استفاده کنیم.

* مثال:
  + "running" (verb) → "run", "better" (adjective) → "good"
  + "رفت#رو" → "می‌روم"

5. Formalizer (رسمی ساز)

رسمی ساز: به تبدیل متن غیر رسمی یا محاوره ای به سبک رسمی تر اشاره دارد. اغلب در برنامه هایی مانند رسمی کردن پیام های متنی، پست های رسانه های اجتماعی یا سخنرانی غیررسمی استفاده می شود.

* به عنوان مثال: "wanna" → "want to", "gonna" → "going to"

6. Normalizer (نرمال ساز)

نرمال ساز: عادی سازی شامل استاندارد کردن متن به یک قالب ثابت است که شامل تبدیل متن به حروف کوچک، حذف علائم نگارشی، گسترش انقباضات و تصحیح املا است.

* مثال: "I've gotta go!" → "i have got to go"

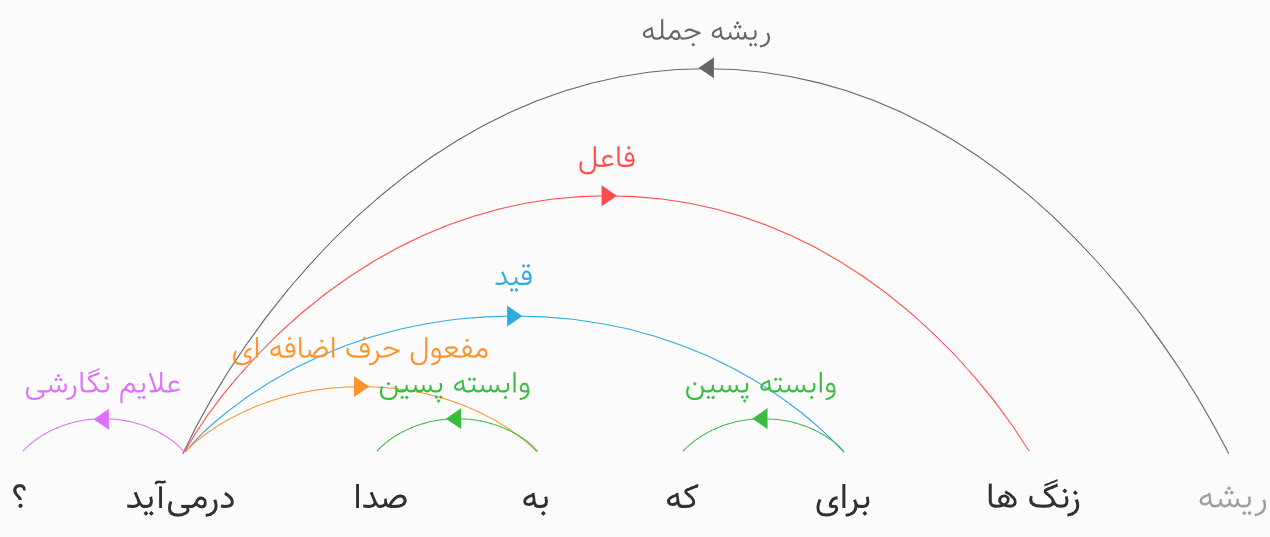
7. Parser (تجزیه کننده)

تجزیه کننده: تجزیه کننده ساختار نحوی یک جمله را با توجه به دستور زبان داده شده تجزیه و تحلیل می کند. جملات را به اجزای آنها، مانند عبارات، بندها و بخش هایی از گفتار تجزیه می کند.

* مثال: تجزیه "روباه قهوه ای سریع از روی سگ تنبل می پرد" فاعل، فعل، مفعول و غیره را مشخص می کند.

8. Dependency Parser (تجزیه کننده وابستگی، تجزیه نحوی)

تجزیه نحوی: این تجزیه کننده نه تنها ساختار نحوی را مشخص می کند، بلکه روابط (وابستگی) بین کلمات یک جمله را نیز مشخص می کند و نشان می دهد که کدام کلمات به دیگران وابسته هستند.

* مثال‌ها:
  + "The quick brown fox jumps over the lazy dog."
  + وابستگی ها:
    - "jumps" ریشه است.
    - «fox» (subject) به «jumps» بستگی دارد.
    - «over» (preposition) به «jumps» بستگی دارد.
    - "dog" (object of the preposition) به "over" بستگی دارد.
  + "زنگ ها برای که به صدا در می‌آیند؟"
  + درخت تحلیل نحوری:
    - 

9. Embedder / Word Embedder

Embedder / Word Embedder: اینها به تکنیک هایی اشاره می کنند که کلمات یا متن را به نمایش های برداری پیوسته (embeddings) تبدیل می کند که معنای معنایی را به تصویر می کشد. تکنیک های رایج عبارتند از Word2Vec، GloVe و BERT.

* مثال: "شاه" و "ملکه" ممکن است به بردارهایی تبدیل شوند که شباهت معنایی آنها را منعکس می کنند.

\*\*\* Embedder میتواند در سطح حرف(character)، کلمه(word) و یا جمله(sentence) نیز عمل کند ولی معمولا بیشتر از embedder در سطح کلمه استفاده میکنیم.

10. Tagger (برچسب زدن)

برچسب‌گذار: به معنای وسیع‌تر، یک برچسب‌گذار برچسب‌ها را به عناصر یک متن اختصاص می‌دهد. متداول‌ترین انواع شامل برچسب‌های POS (تخصیص تگ‌های بخشی از گفتار یا تحلیل صرفی) و برچسب‌های شناسایی موجودیت نام‌دار (NER) (شناسایی نهادهایی مانند نام‌ها، تاریخ‌ها، مکان‌ها) است.

* مثال (NER): "باراک اوباما در هاوایی به دنیا آمد."
  + برچسب ها: [باراک اوباما]/PERSON در [هاوایی]/LOCATION متولد شد.

هر یک از این توابع در NLP برای پیش پردازش، تجزیه و تحلیل، و استخراج اطلاعات معنی دار از متن ضروری است و کارهای پیشرفته تری مانند ترجمه ماشینی، خلاصه سازی متن و تجزیه و تحلیل احساسات را امکان پذیر می کند.

## سوال چهارم

کد و توضیحات مربوط به حل این سوال در قسمت `#Q4` نوتبوک `Harf\_Hazm\_workshop.ipynb` که در پیوست آمده است قابل مشاهده است. توجه کنید برای آماده سازی بستر اجرای کد‌ها باید قسمت `#setting up files, packages, and libraries ` اول اجرا شود.

## سوال پنجم

کد و توضیحات مربوط به حل این سوال در قسمت `#Q5` نوتبوک `Harf\_Hazm\_workshop.ipynb` که در پیوست آمده است قابل مشاهده است. توجه کنید برای آماده سازی بستر اجرای کد‌ها باید قسمت `#setting up files, packages, and libraries ` اول اجرا شود.

## سوال ششم

کد و توضیحات مربوط به حل این سوال در قسمت `#Q6` نوتبوک `Harf\_Hazm\_workshop.ipynb` که در پیوست آمده است قابل مشاهده است. توجه کنید برای آماده سازی بستر اجرای کد‌ها باید قسمت `#setting up files, packages, and libraries ` اول اجرا شود.

## سوال هفتم

کد و توضیحات مربوط به حل این سوال در قسمت `#Q7` نوتبوک `Harf\_Hazm\_workshop.ipynb` که در پیوست آمده است قابل مشاهده است. توجه کنید برای آماده سازی بستر اجرای کد‌ها باید قسمت `#setting up files, packages, and libraries ` اول اجرا شود.

مراجع:

<https://chatgpt.com/>   
<https://www.roshan-ai.ir/hazm/>   
<https://www.roshan-ai.ir/harf/>   
<https://huggingface.co/spaces/evaluate-metric/wer>

پایان