درس پردازش گفتار

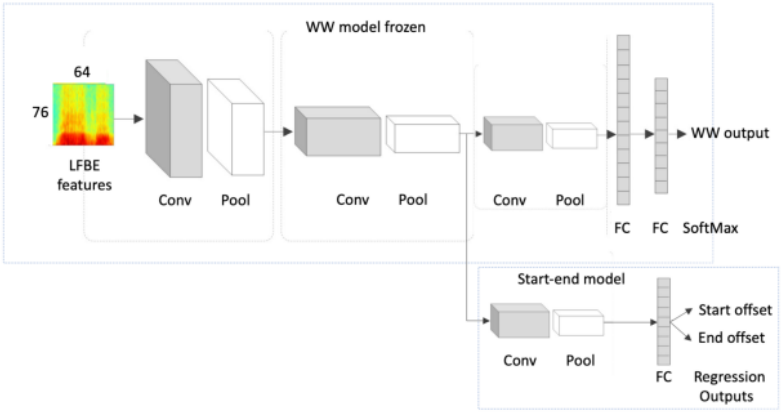
تمرین اول

محمد جواد رنجبر

۸۱۰۱۰۱۱۷۳

سوال ۱

- کاربردی که در گوشی‌های هوشمند هست تشخیص یک سری از کلمات هست که به آن‌ها Wake word گفته می‌شود، در مقاله‌ی Accurate Detection of Wake Word Start and End Using a CNN دو روش جدید را با استفاده از یک شبکه عصبی کانولوشنال (CNN) برای تشخیص دقیق شروع و نقاط پایانی wake word در یک جریان ثابت صدا پیشنهاد می‌کند.



تصویر ۱تشخیص wake word

مدل پیشنهادی به دقت بالایی در تشخیص نقاط پایانی کلمه بیداری، با خطای استاندارد کمتر از ۵۰ میلی ثانیه در مقایسه با داده های برچسب زده شده توسط انسان، دست یافت. این عملکرد با روش‌های مرسوم که بر ترکیبی از مدل‌های آکوستیک و مدل‌های پنهان مارکوف تکیه دارند، همتراز است.

- یک کاربرد پردازش صوت در تولید صوت با صدای اشخاص است که می‌توان در کاربردهای مختلفی مانند: دوبله، ترجمه گفتار به گفتار، تولید صوت برای افرادی که نمی‌توانند صحبت کنند، دستیار صوتی و... استفاده کرد.

گزارش کوتاه از مقاله‌ی Voicebox: Text-Guided Multilingual Universal Speech Generation at Scale:

یکی از مدل‌های قوی و معروف در این تولید صوت، مدل Voicebox است که توسط متا[1] ساخته شده است. Voicebox یک مدل تطبیق جریان غیر اتورگرسیو است که برای تکمیل گفتار، با توجه به صوت ورودی و متن آموزش داده شده است، و بر روی بیش از ۵۰ هزار ساعت گفتار آموزش داده شده است.

مشابه GPT، Voicebox می‌تواند بسیاری از وظایف مختلف را از طریق یادگیری درون context انجام دهد، به عبارتی با توجه به context صوت ورودی، صوت خروجی را تولید می‌کند که باعث کاربردهایی مانند همانندسازی صدا خواهد شد. Voicebox را می توان برای سنتز متن به گفتار بدون شات تک یا چند زبانه، حذف نویز، ویرایش محتوا، تبدیل سبک و تولید نمونه های متنوع استفاده کرد. به طور خاص، Voicebox از پیشرفته‌ترین مدل TTS صفر شات مدل VALL-E هم در درک (۵.۹٪ در مقابل ۱.۹٪ نرخ خطای کلمه) و هم شباهت صوتی (۰.۵۸۰ در مقابل ۰.۶۸۱) عملکرد بهتری دارد در حالی که تا ۲۰ برابر سریع‌تر است.

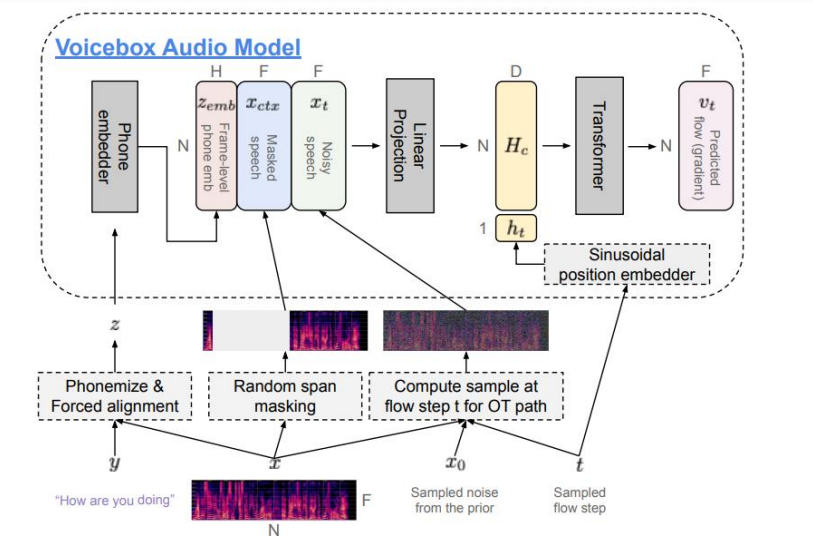
Model	ZS TTS	Denoise	Partial Edit	Sampling

[1] Meta

VALL-E	✓	✗	✗	✓
YourTTS	✓	✗	✗	✓
A3T	✓	✓ (short)	✓	✗
Demucs	✗	✓	✗	✗
Voicebox	✓	✓ (short)	✓	✓

شکل ۱ توانایی‌های VoiceBox

قابلیت‌های این مدل در مقابل مدل‌های معروف قبلی در جدول شماره ۱ نمایش داده شده است. ایده کلی آموزش این مدل مانند مدل‌های بزرگ زبانی یا bert است که با حذف بخشی از جمله سعی می‌کند با توجه به کلمات اطراف آن کلمه را حدس بزند. در این مدل بخشی از سیگنال صوتی را با نویز جایگزین می‌کنند و این مدل‌ها تلاش با condition شدن روی متن و اطراف سیگنال صوتی تلاش در برگرداندن آن بخش می‌کنند به صورتی که context حفظ شود.



تصویر ۲ معماری مدل

عملکرد این مدل در متن در وظیفه تولید متن به صوت بدون شات در مقایسه با سایر مدل‌ها در جدول .. قابل مشاهده است.

Model	WER	SIM-o	SIM-r	QMOS	SMOS
Ground truth	2.2	0.754	n/a	3.98±0.14	4.01±0.09
A3T (cross-sentence)	63.3	0.046	0.146	-	-
YourTTS (cross-sentence)	7.7	0.337	n/a	3.27±0.13	3.19±0.14

VALL-E (cross-sentence)	5.9	-	0.580	-	-
VB-En (cross-sentence)	1.9	0.662	0.681	3.78±0.10	3.71±0.11
A3T (continuation)	18.7	0.058	0.144	-	-
VALL-E (continuation)	3.8	0.452∗	0.508	-	-
VB-En (α = 0.7, continuation)	2.0	0.593	0.616	-	-

شکل ۲ عملکرد Voicebox

از آنجا که فقط به وظیفه تولید صوت از متن پرداختیم، جدول‌های دیگر قابلیت‌های این مدل در مقاله‌ی مربوط به آن قابل مشاهده است و در اینجا نمی‌آوریم.

- یک کاربرد دیگر از کاربردهای پردازش گفتار در ترجمه گفتار به گفتار[۲] می‌باشد.

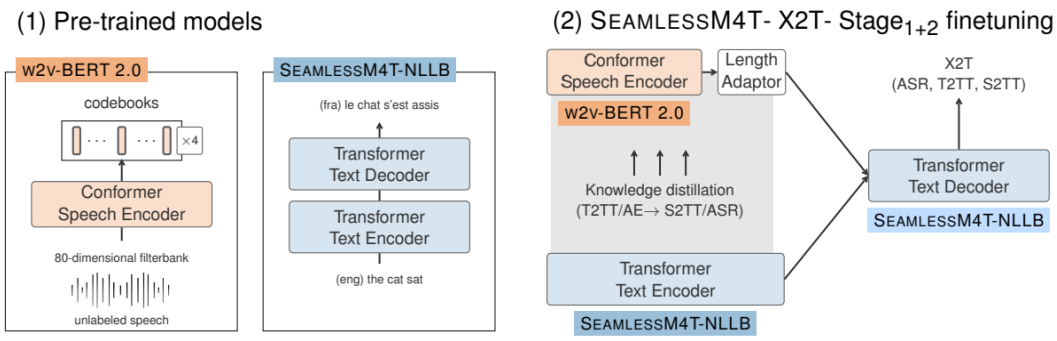
گزارش کوتاه از مقاله‌ی SeamlessM4T: Massively Multilingual & Multimodal Machine Translation:

مدل (Seamless) متا در واقع مجموعه‌ای از مدل‌های هوش مصنوعی است که برای طبیعی‌تر و واقعی‌تر کردن برقراری ارتباط بین زبان‌ها طراحی شده است. این مدل با تمرکز بر ترجمه همزمان و در عین حال حفظ ظرایف گفتار به این هدف دست می‌یابد. قابلیت چندحالته SeamlessM4T :ترجمه گفتار به گفتار، گفتار به متن، متن به گفتار و متن به متن را پشتیبانی می‌کند.

داده و نمایش: این مدل از ۱ میلیون ساعت داده صوتی گفتاری باز استفاده می‌کند تا با استفاده از w2v-BERT 2.0 بتواند representation خوبی یاد بگیرد.

سیستم چندزبانه: در این مقاله یک مجموعه داده موازی و در مودال‌های[۳] مختلف ترجمه گفتاری و متن ایجاد می‌کنند. آن‌ها این مجموعه داده را با داده‌های انسانی و داده‌های نیمه‌انسانی ترکیب می‌کنند تا اولین سیستم چندزبانه را که قادر به ترجمه هم گفتاری و هم متن به و از انگلیسی است، توسعه دهند.

[2] speech-to-speech translation
[3] Multi-modal



تصویر ۳ معماری Seamless x2T

یک معماری کلی از مدل Seamless x2T به در شکل قابل مشاهده است.

عملکرد:

در مجموعه داده FLEURS، SeamlessM4T بهبود ۲۰ درصدی BLEU نسبت به حالت پیشین در ترجمه گفتار به متن دارد .نسبت به مدل‌های دیگر، SeamlessM4T کیفیت ترجمه را در ۱.۲ متریک BLEU در صوت به متن و ۲.۶ BELU در صوت به صوت ارتقا داده است.

سوال ۲

الف)

برای این بخش از وبسایت SpeechTexter.com استفاده کردیم، که نتایج آن در جدول شماره ۱ قابل رویت می‌باشد:

#	جمله‌ی ورودی	خروجی ASR	WER
1	من و خواهرم قصد خرید لباس از فروشگاه اوپال تهران را داشتیم.	من و خواهرم قصد خرید لباس از فروشگاه اوپال تهران را داشتیم.	۰٪
2	من به کاشان رفتم تا روند درست کردن گلابی را ببینم	من به کاشان رفتم تا روند درست کردن گلابی را ببینم	۰٪
3	I have a friend that works in Apple and he really likes his job	I have a friend that works in apple and he really likes his job	۰٪
4	I'm contemplating whether to attend the conference next week or focus on my ongoing projects	I'm contemplating whether to attend the conference next week or focus on my ongoing projects	۰٪
5	She sells seashells by the seashore	she sells seashells by the seashore	۰٪
6	من به نظرم اونا خوداشون اصلاً نمی‌خوان این کارو انجام بدن	من به نظرم اونا خداشون اصلاً نمی‌خوان این کارو انجام بدن	۹٪

۷	ببین ماشین لرنینگ آسونه دیپ لرنینگم آسونه ولی به نظرم برو ان ال پی بردار	ببین ماشین لرنینگ آسونه دیپ لرنینگم آسونه ولی به نظرم برو ان ال پی بردار	۰٪
۸	ببین به نظرت چرا تو فارسی خواهر رو خواهر می‌نویسن	ببین به نظرت چرا تو فارسی خواهر و خواهر می‌نویسن	۱۰٪

شکل ۳ نتایج جملات داده شده به سیستم

- جمله‌ی شماره‌ی ۳ با وجود صحیح بودن از لحاظ املایی، دارای این مشکل می‌باشد که Apple اسم کمپانی و یک اسم خاص است، با این حال مدل با حروف اول کوچک آن را نوشته است.
- جمله‌ی شماره‌ی ۶ به خاطر تلفظ بد گوینده که جمع کلمه‌ی خودهاشون را خداشون گفت، مدل نتوانست آن را درست تشخیص دهد و آن را خداشون خواند.
- جمله‌ی شماره‌ی ۷ از آنجا که خواهر با حرف "ر" تمام می‌شد و سرعت گویش زیاد بود، مدل بخش "ر" از کلمه‌ی "رو" را خوب تشحیص نداده و فقط "و" آن را نوشته.

به طور کلی سیستم نسبتا Robustای بود و با جملات مختلفی که تست کردم اگر قصد به اشتباه انداختن سیستم نداشتم(تلفظ نامفهوم) سیستم تقریبا همه را درست تشخیص داد.

توضیح درباره‌ی Word Error Rate(WER) و چرایی رخ داد آن:

این متریک به معنای نرخ خطا در کلمات است. این یک معیار رایج برای ارزیابی عملکرد سیستم‌های تشخیص گفتار خودکار(ASR) ، سیستم‌های ترجمه ماشینی و سایر وظایف پردازش زبان است WER .اختلاف بین متن اصلی یا متن مرجع و خروجی تولید شده توسط سیستم مورد ارزیابی را اندازه‌گیری می‌کند.

WER با شمارش تعداد خطاها در خروجی سیستم، از جمله جایگزینی‌ها، درج‌ها و حذف‌ها، محاسبه می‌شود و سپس این تعداد را بر تعداد کل کلمات در متن مرجع تقسیم می‌کند. نتیجه به عنوان یک درصد عبارت از تفاوت در دقت نمایش داده می‌شود، که مقادیر کمتر WER به دقت بهتر اشاره دارند.

فرمول محاسبه WER به صورت زیر است:

$$[\mathrm{WER} = \frac{\mathrm{S} + \mathrm{I} + \mathrm{D}}{\mathrm{N}} \times 100]$$

که:

- S: تعداد جایگزینی‌ها (کلماتی که به‌طور جایگزین می‌شوند) است.
- I: تعداد درج‌ها (کلمات اضافی که توسط سیستم به صورت نادرست قرار داده می‌شوند) است.
- D: تعداد حذف‌ها (کلمات موجود در متن مرجع که در خروجی سیستم وجود ندارند) است
- N: تعداد کل کلمات در متن مرجع است.

هدف از این که WER را کمینه کنیم، بهبود دقت سیستم در تصویرسازی یا ترجمه زبان گفتاری یا نوشتاری است.

دلایل اتفاق افتادن خطا در یک سیستم بازشناسی گفتار موارد زیر می‌تواند باشد:

- مشکلات کیفیتی سیستم:

- کیفیت پایین صوت گوینده
- تلفظ بد، یا نادرست: در سیستم‌هایی که به اندازه کافی قوی نیستن لهجه‌های مختلف گوینده یا تلفظ‌های آن شخص ممکن است باعث ایجاد خطا شود.
- نویز یا اختشاش در صدای ورودی
- کلامت Out Of Vocabulary (OOV): اسامی و کلامات خاصی که مثلا در داده‌های آموزش ورودی نبوده‌اند و یا خیلی کم استفاده می‌شوند.

- مشکلات زبانی:
  - هم‌خوان‌ها: وجود حرفی مانند: ص،ث و س در زبان فارسی ممکن است باعث چالش در این سیستم‌ها و غلط‌های املایی شوند.
  - کلمات نزدیک به هم: با توجه به سیستم و اینکه ممکن است توانایی درک جملات context را نداشته باشد، کلماتی مانند Bare، Bear ممکن است توسط سیستم اشتباه تشخیص داده شوند. همچنین کلماتی مانند گل آبی یا گلابی را ممکن است سیستم بهم بچسباند.

ب) WER در جدول قسمت الف موجود است با این حال محاسبه‌ی در این بخش نشان داده می‌شود:

#	WER
۱	$\text{WER} = \frac{\text{S}+\text{I}+\text{D}}{\text{N}} \times 100\% = \frac{0}{12} \times 100\% = 0\%$
۲	$\text{WER} = \frac{\text{S}+\text{I}+\text{D}}{\text{N}} \times 100\% = \frac{0}{11} \times 100\% = 0\%$
۳	$\text{WER} = \frac{\text{S}+\text{I}+\text{D}}{\text{N}} \times 100\% = \frac{0}{14} \times 100\% = 0\%$
۴	$\text{WER} = \frac{\text{S}+\text{I}+\text{D}}{\text{N}} \times 100\% = \frac{0}{15} \times 100\% = 0\%$
۵	$\text{WER} = \frac{\text{S}+\text{I}+\text{D}}{\text{N}} \times 100\% = \frac{0}{6} \times 100\% = 0\%$
۶	$\text{WER} = \frac{\text{S}+\text{I}+\text{D}}{\text{N}} \times 100\% = \frac{1}{11} \times 100\% = 9\%$
۷	$\text{WER} = \frac{\text{S}+\text{I}+\text{D}}{\text{N}} \times 100\% = \frac{0}{15} \times 100\% = 0\%$
۸	$\text{WER} = \frac{\text{S}+\text{I}+\text{D}}{\text{N}} \times 100\% = \frac{1}{10} \times 100\% = 10\%$

شکل ۴ محاسبهWER

پ)

اگر جمله‌ی "Open the Firefox, no sorry, the chrome" را به وبسایت https://www.speechtexter.com/ یا سیستم‌های ASR دیگر بدهیم، از آنجا این سیستم فقط قصد تشخیص کلمات را دارد و سیستمی راجع به اصلاح کلمات به واسطه کلماتی مانند no و اینگونه موارد وجود ندارد همان جمله‌ی "Open the Firefox, no sorry, the chrome" را چاپ خواهد کرد. حال این موضوع را با سیستم‌های چت‌بات، امتحان می‌کنیم، در اینجا فقط دسترسی به google assistant داشتیم بنابراین از این چت‌بات استفاده کردیم.

با گفتن "Open the Firefox, no sorry, the chrome" این سیستم Google Chrome را باز خواهد کرد، این سیستم‌ها که مبتنی بر intent detection هستند می‌فهمند که intent این جمله باز کردن chrome بوده و این کار را انجام می‌دهند. همچنین فیلم این مکالمه در فولدر تکلیف موجود است.

سوال ۳

کد کامل شده در فولدر تکلیف موجود می‌باشد.

سوال ۴

الف)

برای این کار باید فایل‌های صوتی مربوط به موارد زیر را ضبط کنیم:

۱- همه‌ی اعداد بین ۰ تا ۱۹
۲- اعداد ۲۰، ۳۰، ۴۰، ۵۰، ۶۰، ۷۰، ۸۰، ۹۰
۳- اعداد ۲۰، ۳۰، ۴۰، ۵۰، ۶۰، ۷۰، ۸۰، ۹۰ به همراه صدای "اوُ" در انتهای آن‌ها (مانند بیستوُ یک و...)
۴- اعداد ۱۰۰، ۲۰۰، ۳۰۰، ۴۰۰، ۵۰۰، ۶۰۰، ۷۰۰، ۸۰۰، ۹۰۰
۵- اعداد ۱۰۰، ۲۰۰، ۳۰۰، ۴۰۰، ۵۰۰، ۶۰۰، ۷۰۰، ۸۰۰، ۹۰۰ به همراه صدای "اوُ" در انتهای آن‌ها (مانند صدوُ یک و...)
۶- عدد ۱۰۰۰
۷- عدد ۱۰۰۰ به همراه صدای "اوُ"
۸- عدد ۱۰۰۰۰۰۰
۹- عدد ۱۰۰۰۰۰۰ به همراه صدای "اوُ"

جدول زیر تعداد کامل اعدادی که نیاز به ضبط هست را نشان داده‌است:

اعداد	تعداد
۰ تا ۱۹	۲۰
۲۰، ۳۰، ۴۰، ۵۰، ۶۰، ۷۰، ۸۰، ۹۰	۸
۲۰، ۳۰، ۴۰، ۵۰، ۶۰، ۷۰، ۸۰، ۹۰ به همراه "اوُ"	۸
۱۰۰، ۲۰۰، ۳۰۰، ۴۰۰، ۵۰۰، ۶۰۰، ۷۰۰، ۸۰۰، ۹۰۰	۹
۱۰۰، ۲۰۰، ۳۰۰، ۴۰۰، ۵۰۰، ۶۰۰، ۷۰۰، ۸۰۰، ۹۰۰ به همراه "اوُ"	۹
۱۰۰۰۰	۱
۱۰۰۰۰ به همراه "اوُ"	۱
۱۰۰۰۰۰۰	۱
۱۰۰۰۰۰۰ به همراه "اوُ"	۱
مجموع	۵۸

شکل ۵ کلمات خوانده شده

در نتیجه نیاز به ۵۸ فایل صوتی داریم.

البته در صورتی که نخواهیم از صدای "وُ" استفاده کنیم و فقط از "وَ" استفاده کنیم تعداد کمتری صوت برای ضبط نیاز است، همچنین می‌توانیم از "وُ" به گونه‌ای استفاده کنیم که به صورت جداگانه، به صوت‌هایی که نیاز هست متصل شود(که البته باعث غیر طبیعی‌تر شدن خوانش اعداد می‌شود).

بنابراین جدول به صورت زیر تغییر می‌کند:

اعداد	تعداد
۰ تا ۱۹	۲۰
۲۰، ۳۰، ۴۰، ۵۰، ۶۰، ۷۰، ۸۰، ۹۰	۸
۱۰۰، ۲۰۰، ۳۰۰، ۴۰۰، ۵۰۰، ۶۰۰، ۷۰۰، ۸۰۰، ۹۰۰	۹
وَ یا وُ	۱
۱۰۰۰۰	۱
۱۰۰۰۰۰۰	۱
مجموع	۴۰

شکل ۶ کلمات خوانده شده(ورژن کوتاهتر)

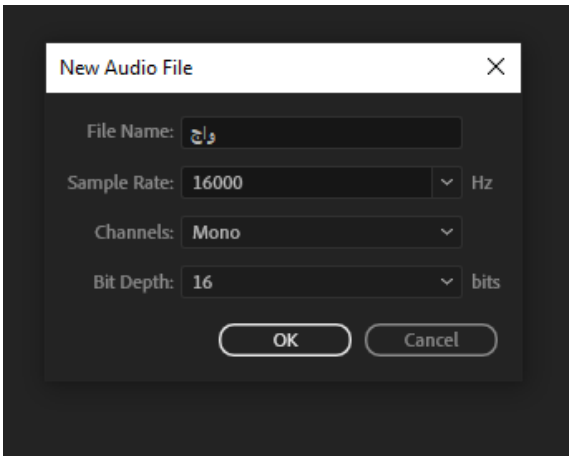
برای این حالت به ۴۰ فایل صوتی نیاز داریم.

حال دوباره اگر واقعا قصد کم کردن تعداد فایل صوتی را داریم، برای اعدادی شامل ۳۰۰، ۴۰۰، ۵۰۰، ۶۰۰، ۷۰۰، ۸۰۰، ۹۰۰ می‌توان با ترکیبات زیر این اعداد را خواند و تعداد صوت کمتری ضبط کرد.

سیصد= سی+صد | چهارصد=چهار+صد | ششصد=شش+صد | هفتصد=هفت+صد | هشتصد=هشت+صد | نهصد=نه+صد

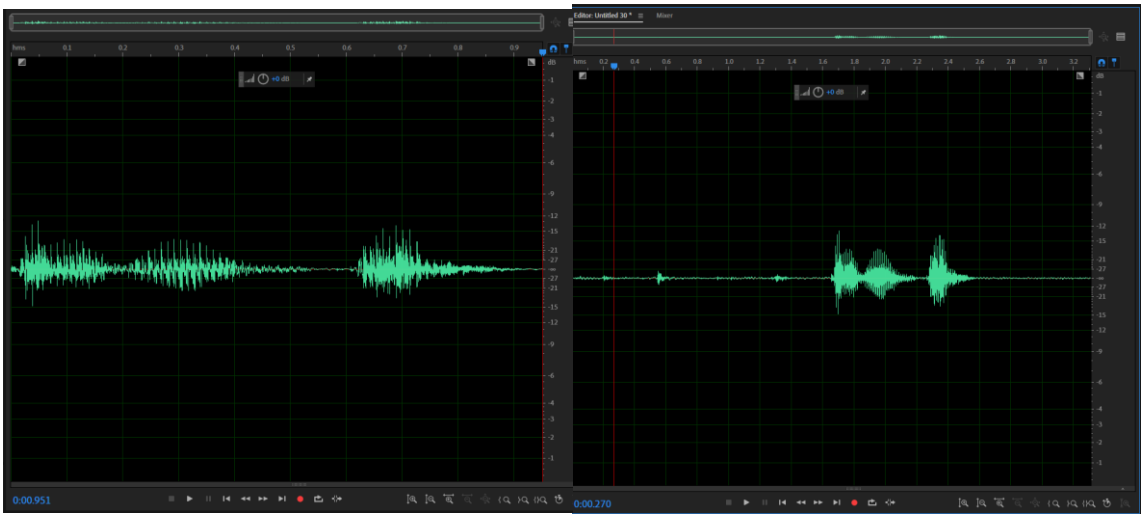
با استفاده از این نحوه نیز ۱۲ فایل صوتی کمتر نسبت به مدل اول ضبط خواهیم کرد (46 صوت) و نسبت به حالت دوم ۶ فایل صوتی کمتر ضبط خواهیم کرد(۳۴ صوت). در کد نیز منطق بالا استفاده شده ولی همه‌ی فایل‌های صوتی مورد نیاز ضبط شده است.

حال ضبط کردن صوت‌ها را شروع می‌کنیم، برای این کار از برنامه‌ی Adobe Audition استفاده می‌کنیم، ابتدا پروژه‌ی جدیدی ساخته و شروع به ضبط صوت با استفاده از تنظیمات خواسته شده در سوال می‌کنیم.



تصویر ۴ تنظیمات ضبط صوت

بعد از ضبط صوت نیز بخش‌های سکوت و نویز ابتدا و انتهای صوت را حذف می‌کنیم(البته با کد هم می‌توانستیم اینکار را انجام دهیم)، و صوت را ذخیره می‌کنیم.



تصویر ۵ حذف بخش‌های اضافی

بعد از ضبط صوت‌ها به کد می‌پردازیم.

از آنجا که قصد داریم هم کلمات فارسی و هم اعداد ریاضی را بخوانیم یک دیکشنری تعریف کردیم که کلمات فارسی را به فایل مورد نظر map کرده و آن فایل‌های صوتی را به یکدگیر می‌چسبانیم و خوانش یک عدد خواهد بود.

```
def string_to_audio(text,formal):
  if (not formal):
    text=text_preprocess(text)
  tokenized = text.split()
  audio_files=[]
  for digit in tokenized:
    audio_files.append(digit+".wav")
  result_audio = concatenate_audio_files(audio_files)
  return result_audio
```

کد بالا مراحل انجام اینکار را نشان می‌دهد ابتدا در صورتی که بخواهید از "وَ" به جای "وُ" استفاده کنیم از فانکشن text_preprocess استفاده می‌کنیم، (البته می‌توانیم پیش پردازش‌های دیگری نظری تصحیح غلط تایپی و اینجور موارد انجام دهیم) سپس را split کرده و با فاصله از هم جدا می‌کنیم، و فایل مربوط به هر صوت را خوانده و به ترتیب بهم وصل می‌کنیم.

برای تبدیل اعداد به متن نیز از int_to_fa استفاده می‌کنیم، که به این صورت کار می‌کند:

تابع ابتدا یک دیکشنری به نام d تعریف می‌کند تا اعداد را به واژه‌های فارسی متناظر آنها نگاشت کند. این دیکشنری شامل نگاشت‌ها برای اعداد یک رقمی، ضرب‌هایی از ۱۰ تا ۹۰ و نگاشت‌های خاص برای اعداد تا ۲۰ است.

ثابت‌های k ، m، b و t برای نمایش هزار، میلیون، میلیارد و تریلیون به ترتیب تعریف شده‌اند.

- تابع سپس مقدار ورودی num را با این ثوابت مقایسه کرده.
- اگر num کمتر از ۲۰ باشد، واژه متناظر از دیکشنری را برمی‌گرداند.

- اگر num کمتر از ۱۰۰ باشد، بررسی می‌کند که آیا ضریبی از ۱۰ است یا خیر و بر اساس این موضوع تبدیل به واژه‌های فارسی را انجام می‌دهد.
- اگر num کمتر ۱۰۰۰ باشد، صدگان را انجام می‌دهد و تبدیل بخش باقی‌مانده را به یک فراخوانی بازگشتی واگذار می‌کند.
- همین الگو برای مقادیر کمتر ازm ، bو t ادامه پیدا می‌کند.

بنابراین بعد از تبدیل اعداد به متن، دوباره تابع مربوط به متن را صدا می‌زنیم و فرایند مروبط به آن را طی می‌کنیم.

صوت‌های خواسته شده شامل ۱۴۰۹ و دویست و دو هزار و ده تولید شده و در فایل تمرین موجود است.

ب)

برای این بخش نیز مانند بخش قبل عمل می‌کنیم و تمام واج‌ها را ضبط می‌کنیم و در فولدر مربوط قرار دادیم. حال تمام کلماتی که می‌خواهیم آن را بخوانیم با اعراب آن‌ها به عنوان ورودی می‌دهیم و در خروجی صوت‌های تولیدی را دریافت می‌کنیم.

از آنجایی که این صوت‌ها از بخش‌های کوچکتری تشکیل شده‌اند پس سکته‌های بیشتری در آن‌ها رخ می‌دهد که کیفیت آن‌ها را تا حد زیادی کاهش می‌دهد.

بنابراین با مقیاسه صوت‌های ۱۴۰۹ و دویست و دو هزار و ده با بخش قبل کاهش کیفیت نسبت به بخش الف دارد و تا حدی نا مشخص است.

حال تمام صوت‌های خواسته شده در صورت سوال را تولید می‌کنیم و در فایل مربوطه قرار می‌دهیم.

برای قابل فهم‌تر شدن این کلمات می‌توان از روش‌های زیر استفاده کرد:

- Edit صوت‌ها:
  - Overlap: افزودن overlap تا قبل از پایان صوت واج قبلی صوت واج بعدی شروع شود و حس حرف ممتد را ارائه دهد.
  - حذف سکوت: حذف کردن سکوت از صوت‌ها که اینکار در هنگام ضبط انجام شده ولی می‌توان در کد نیز از یک آستانه‌ای به پایین رو حذف کرد.
  - Fading: افزودن fade in و fade out به صوت‌ها تا بیشتر شبیه حرف زدن واقعی شود.
  - افزودن سکوت: با افزودن سکوت بین هر کلمه می‌توان فهم آن کلمه را بیشتر کرد.
- ضبط صوت‌های بهتر
  - هم‌خوان+واکه: می‌توانیم برای بهتر شدن صوت‌ها تمام هم‌خوان‌ها را کنار واکه‌ها ضبط کنیم تا سکته‌های کمتری در صوت باشد و صوت طبیعی‌تر باشد برای مثال در کلمه‌ی مُحَمَمَد صوت‌های ضبط شده شامل: "مُ"، "حَ"، "م"، "مَ"، "د" باشد.