دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکده مهندسی برق و
کامپیوتر

پیاده‌سازی تبدیل گفتار به متن بر روی میکروکنترلر

پایان‌نامه برای دریافت درجه کارشناسی
در رشته مهندسی برق گرایش سیستم‌های دیجیتال

نام

سید محمد حسینی

شماره دانشجویی

۸۱۰۱۹۴۵۴۱

استاد راهنما:

دکتر مهدی کمال

بهمن ماه ۱۳۹۸


بسم الله الرحمن الرحيم

تعهدنامه اصالت اثر
باسمه تعالی

اینجانب سید محمد حسینی تائید می کنم که مطالب مندرج در این پایان نامه حاصل کار پژوهشی اینجانب است و به دستاوردهای پژوهشی دیگران که در این نوشته از آنها استفاده شده است مطابق مقررات ارجاع گردیده است. این پایان نامه قبلاً برای احراز هیچ مدرک هم سطح یا بالاتر ارائه نشده است.
کلیه حقوق مادی و معنوی این اثر متعلق به دانشکده فنی دانشگاه تهران می باشد.

نام و نام خانوادگی دانشجو : سید محمد حسینی

امضای دانشجو :

تقدیم[1] به:

تقدیم به مادر و پدر عزیزم.

این پروژه گوشه‌ای از نتایج زحمات بی‌شمار آنهاست.

[1] Dedication

تشکر و قدردانی[1]:

با تشکر از شرکت هوما، که با کمک و همراهی آنها این پروژه به ثمر نشست.

[1] Acknowledgements

چکیده[1]

دستیاران صوتی در حال حاضر یکی ازموضوعات بسیار داغ و جذاب جهان می‌باشد. این سیستم‌های هوش مصنوعی، با دریافت گفتار انسان و تشخیص آن میتوانند به عنوان یک راهنما و دستیار در کنار انسان بوده و رنج وسیعی از نیازهای انسان را برطرف کنند.

این دستیاران با دریافت صدای انسان و ارسال آن به سرورهای ابری شرکت سازنده سعی در تبدیل گفتار به متن و سپس پردازش زبان‌های طبیعی آن دارند. این دستگاه‌ها برای دریافت صدا همیشه باید آماده باشند و هرگونه صدا را به سرور ابری شرکت ارسال کنند که این عمل از نظر توان مصرفی بهینه نیست و سیستم توان بسیاری مصرف می‌کند. از سمتی دیگر، دریافت تمام اصوات موجود در محیط و ارسال آن‌ها به سرور ابری باعث ایجاد مشکلات بسیاری در جهت حفظ حریم خصوصی کاربران می‌شود.

برای رفع مشکلات موجود، شرکت‌های بزرگ کلمه‌ای را تحت عنوان کلمه‌ی داغ در سیستم تعبیه میکنند که دستگاه با دریافت آن از حالت نیمه خواب بیدار شده و سپس اطلاعات دریافتی را به سرور ابری ارسال می‌کند. این کلمه‌ی داغ به صورت غیر برخط[2] پردازش می‌شود.

سیستم‌های موجود بر روی پردازنده‌های نسبتا قوی مانند پردازنده‌های تلفن همراه پیاده سازی شده است که در سیستم‌های نهفته تقریبا توان مصرفی متوسطی دارند. به همین جهت در یکسال اخیر میکروکنترلرها که توان مصرفی کمتری نسبت به آن دسته از پردازنده‌ها دارند محبوب شده‌اند و مهندسان سعی در ایجاد سیستم‌های نهفته و اینترنت اشیا بر روی میکروکنترلرها دارند.

در این پروژه با استفاده از ویژگی‌های جدید معرفی شده توسط شرکت‌های بزرگ، سیستم تشخیص کلمات داغ بر روی میکروکنترلر پیاده سازی، تست شده و نتیجه نهایی دریافت شده است. این سیستم از توان مصرفی بسیار کمی برخودار است که باعث می‌شود که بسیار اقتصادی باشد.

کلمات کلیدی:

میکروکنترلر، گفتار، کلمات داغ، تشخیص گفتار، شبکه‌های عصبی، یادگیری ماشین، سیستم‌های نهفته

[1] Abstract

[2] Offline

فهرست مطالب

فهرست علائم اختصاری

NN	Neural Networks
uC	Micro Controller
uP	Micro Processor
MFCC	Mel-frequency Cepstral Coefficients

این صفحه سفید است.

فصل ۱

مقدمه و توضیح مسئله

در سال‌های اخیر با توجه در یادگیری عمیق، فضای مناسبی برای تشخیص گفتار که از لحاظ تکنیکال، پیش‌بینی و فهم که مبحث پیچیده‌ای است، فراهم شده است. سیستم‌هایی که بر این پایه ایجاد شده‌اند بسیار قابل اطمینان و با دقت می‌باشند که میتوان با استفاده از آنها سیستم‌های بسیاری از جمله دستیاران صوتی خودروهای خودران و خانه‌های هوشمند را نام برد. دستیاران صوتی با استفاده از تبدیل صوت به متن و سپس به بررسی متن ورودی پرداخته و در صورت نیاز عمل مورد نظر را انجام می‌دهد.

مقدمه

امروزه برای تبدیل گفتار به متن و سپس پردازش زبان طبیعی شرکت‌های بزرگ از سیستم‌های ابری استفاده می‌کنند. صوت ضبط شده از کاربر به سرورهای موجود شرکت‌ها ارسال و پس از بررسی و پردازش، نتایج به سمت کاربر باز می‌گردد. این سیستم علارغم زمان پاسخ گویی مناسب مشکلاتی دارد از جمله:

۱) سیستم به طور مداوم باید به محیط گوش دهد و اصوات را به سرورهای ابری ارسال و پاسخ را دریافت کند.

۲) ضبط مداوم اصوات باعث کاهش حریم شخصی کاربران می‌شود و تمامی اصوات محیطی که دستگاه در آن قرار دارد ضبط به عبارت دیگر شنود می‌شود.

برای رفع مشکل اول سیستم‌هایی با توان پایین را ایجاد شده است که به صورت آفلاین به کلمات داغی[1] حساس می‌باشند و در صورت تشخیص کلمه مورد نظر سیستم از حالت نیمه خاموش [2] خارج شده و اصوات بعدی را دریافت، به سرورها ارسال و نتیجه را بازمیگرداند. ایجاد همچین سیستمی بسیار به کاهش توان مصرفی سیستم و حساسیت سیستم را به اتصال همیشگی به اینترنت کاهش می‌دهد. همچنین دستگاه در صورت تشخص کلمه داغ می‌تواند سیگنال‌های بعدی را ضبط کرده و در صورت اتصال به اینترنت به سرور ارسال کند و پاسخ را دریافت کند.

برای دومین مشکل، در سال‌های اخیر شرکت‌های متن‌باز تلاش کرده‌اند که ابتدا با ایجاد یک سیستم متن باز کلی برای تشخیص گفتار به صورت متن‌باز در معرض عموم قرار دهند که دیگر شرکت‌ها بتوانند از این قابلیت استفاده کنند و انحصار تبدیل گفتار به متن را کاهش دهند. همچنین با معرفی سیستم Tesorflow Lite توسط گروه Tensorflow این امکان فراهم شد که بتوان مدل‌های آموزش داده شده بر روی کامپیوترهای قدرتمند را بر روی سیستم هایی با سرعت پردازشی پایین تر نیز قرار داد. به همین دلیل هم اکنون گروه‌های مختلفی در تلاش هستند که بتوانند مدلی را ارائه دهند که بتوان به صورت آفلاین بر روی دستگاه‌هایی با توان پردازشی پایین‌تر مانند تلفن همراه، دستیاران صوتی و گجت‌های کوچک قرار دهند تا از ارسال صورت به سرورهای ابری جلوگیری کرده و پردازش زبان به صورت آفلاین بر روی دستگاه مورد نظر انجام شود.

۱.۱- تاریخچه‌ای از موضوع تحقیق

تشخیص گفتار تاریخی طولانی‌تر از دستیار شخصی شرکت اپل[3]، سیری[4] که در سال ۲۰۱۱

[1] Hot Words
[2] Idle
[3] Apple
[4] Siri

معرفی شد، دارد. در نمایشگاه سیاتل سال ۱۹۶۲ شرکت IBM معصولی تحت عنوان جعبه کفش[1] معرفی کرد. این جعبه که به اندازه یک جعبه کفش بود میتوانست ۱۶ کلمه مختلف را تشخیص دهد. در سال‌های بعد دانشگاه پنسیلوانیا[2] و دیگر سازمان‌ها بر روی این پروژه کار کردند. در سال ۲۰۱۱ شرکت اپل، سیری را معرفی کرد و در ادامه شرکت گوگل[3] در سال ۲۰۱۲ دستیار حالا گوگل[4]، در سال ۲۰۱۴ شرکت مایکروسافت[5] دستیار کورتانا[6] و در همان سال شرکت آمازون[7] دستیار الکسا[8] را معرفی کردند.

در سال ۲۰۱۴ موتور جست‌وجو بزرگ کشور چین، موسوم به بایدو[9]مقاله‌ای ارائه کردند که در آن به ساختار جدید تحت شبکه‌های عصبی برای تبدیل گفتار به متن ارائه داده بودند. در این مقاله، از جمع آوری مجموعه داده تا پیاده‌سازی شبکه مورد نظر و ارزیابی آن صحبت شده است. این شبکه توسط فریمورک قدرتمند Tensorflow پیاده سازی شده است. این مقاله با استفاده از شبکه‌های عصبی و لایه‌های تمام متصل و بازگشتی، با دقت بسیار بالا گفتار صوتی را به متن تبدیل میکند.

شکل (۱-۱) - زمان‌بندی معرفی دستیاران صوتی[10]

[1]Shoebox
[2]Pennsylvania
[3]Google
[4]Google Now
[5]Microsoft
[6]Cortana
[7]Amazon
[8]Alexa
[9]Baidu
[10]منبع: https://www.smartsheet.com

در سال ۲۰۱۹ گروه Tensorflow، یک ویژگی جدید را تحت Tensorflow Lite Micro معرفی کردند. این ویژگی جدید، با تبدیل شبکه‌ی آموزش داده شده توسط کامپیوتر شخصی را به یک شبکه بسیار کوچک که قابلیت استفاده بر روی یک میکروکنترلر را دارد. این ویژگی دنیای جدید بر روی میکروکنترلرها ایجاد کرد. میکروکنترلرها که به مصرف توان پایین و دارا بودن درگاه‌های ارتباطی معروف هستند. با ورود شبکه‌های عصبی به میکرکنترلرها میتوان گامی بلند به سمت اینترنت اشیاء و گجت‌هایی با توان مصرفی بسیار کم برداشت.

۱.۲- شرح مسئله تحقیق

آشنایی با سیستم‌های Embedded و پیاده سازی شبکه‌های عصبی بر روی آنها بحث بسیار داغی میباشد. چالش‌های موجود از جمله سرعت پردازش پایین، نبود حافظه اصلی بزرگ و ... باعث می‌شود که مهندسان در تلاش باشند که بتوانند شبکه‌های عصبی را بر روی سیستم‌هایی با منابع محدود پیاده سازی کنند.

سیستم مورد نظر از دو بخش کلی نرم‌افزاری و سخت‌افزاری ایجاد شده است. در قسمت نرم‌افزاری، نیاز بر این است که با استفاده از شبکه‌های عصبی، شبکه‌ای را آموزش دهیم که بتواند کلماتی مانند yes, no را تشخیص دهد. این شبکه به به نوعی وظیفه‌ی دسته‌بندی ورودی های صوتی را دارد. این شبکه ورودی‌های یک ثانیه‌ای را دریافت میکند و بر اساس برچسب‌هایی که از قبل به اصوات خورده است، آنها را یاد میگیرد و در ادامه با ورودی جدید میتواند کلمه مورد نظر را تشخیص دهد. با توجه به اینکه ممکن است کلماتی که در سیستم موجود نیستند و یا صدای خالی به سیستم داده شود، نیاز است که ۲ حالت دیگر را هم سیستم بتواند تشخیص دهد که به صورت silence و unknown میباشد. البته این دو مورد در قسمت پیاده‌سازی قرار داده میشوند. سپس نیاز است که شبکه یاد گرفته شده را به صورت یک شبکه با حجم بسیار پایین‌تر و به صورت lite آن را تبدیل کنیم. به همین منظور با استفاده از ویژگی جدید تنسورفلو میتوان این کار را انجام داد و شبکه‌ی سبک‌تر را ایجاد کرد.

در قسمت سخت‌افزاری نیاز است که بتوان جریانی ایجاد کرد که کلمه به مدل یادگرفته شده رسیده و سپس خروجی به یک سیستم دیگر داده شود. برای دریافت صدا از میکروفن استفاده شده و صدای دریافتی به طیف‌سنج تبدیل و سپس به مدل داده می‌شود.

۱.۳- اهداف و آرمان‌های کلی تحقیق

در این مسئله تلاش بر این بود که بتوان یک شبکه‌ی عصبیی را آموزش داد که بر روی

میکروکنترلر قابل پیاده سازی باشد. به همین منظور به دنبال جریان کنونی جهانی و تکنولوژی، نیاز به این صورت دیده شد که یک سیستمی برای تشخیص کلمات داغ برروی میکروکنترلر ایجاد نماییم که به صورت یک MVP برای شروع یک پروژه بزرگ‌تر دستیار خانگی کاملا ساخت کشور عزیزمان ایران باشد. به همین منظور نیاز دیدیم که یک مسیر برای پیاده سازی این مورد را هموار کنیم که در آینده بر پایه آن سیستم اصلی را ساخته و در مقیاس صنعتی تولید و به فروش رساند.

هدف اصلی این پروژه ایجاد یک سیستم برای تشخیص کلمات داغ توسط میکروکنترلر و در ادامه انجام یک پردازش حجیم‌تر مانند ارسال ادامه‌ی صوت به یک سرور ابری و یا پردازش آفلاین صوت می‌باشد.

۱.۴- روش انجام تحقیق

ابتدا برای آشنایی با موضوع تحقیق و پروژه، از مقاله‌های موجود در این مورد مانند مقاله بایدو و سپس پروژه متن‌باز شرکت موزیلا استفاده شده است. مقاله بایدو تمرکز اصلی خود را به ایجاد شبکه و تحلیل شبکه اختصاص داده است. در این مقاله توضیحات کاملی در مورد نحوه‌ی دیدگاه تبدیل گفتار به متن و انواع شبکه‌های مورد نیاز صحبت شده است. در ادامه مقاله، توضیحات پیاده‌سازی شبکه طراحی شده و آزمایش و خطا سنجی آن توضیح داده شده است. در ادامه آن پروژه متن‌باز موزیلا بررسی شد که با دیدگاه پیاده‌سازی شبکه‌های عصبی در حوزه صوت آشناییت پیدا شود. در این پروژه متن باز که مجموعه داده نیز توسط همین شرکت جمع آوری شده است، مقاله بایدو به صورت متن‌باز پیاده شده است که عموم میتوانند از آن استفاده کنند.

در ادامه و آشنایی با این حوزه، برای انجام و پیاده‌سازی پروژه، از اسناد موجود داخل سایت اصلی tensorflow استفاده شده است. این اسناد با دقت بالا بررسی شده و سپس شروع به پیاده ساز شبکه عصبی و سپس پیاده‌سازی بر روی سخت افزرا انجام شد. در این مسیر همراهی و کمک خبرگان این حوزه یکی از علل تسریع انجام پروژه شد. همچنین همراهی پشتیبانی شرکت موزیلا که در رساندن مجموعه داده و همراهی آن‌ها در پاسخ‌گویی به سؤالات این‌جانب کمک شایانی کرد.

۱.۵- ساختار پایان‌نامه

در ادامه، در فصل دوم به بررسی تعاریف و مفاهیم اولیه همچون شبکه‌های عصبی، سیستم‌های نهفته و اجزای اصلی سیستم که به منظور درک عمیق‌تر و آشنایی بیشتر با این حوزه خواهیم داشت. در این فصل به صورت مفصل پیش‌نیازهای لازم برای پیاده سازی این پروژه بحث خواهد شد.

در فصل سوم به توضیح اجزای سیستم خواهیم پرداخت. همانطور که در ادامه خواهید خواند سیستم به دو بخش اصلی آموزش شبکه عصبی و بعد از آن پیاده‌سازی بر روی سخت افزار تفکیک شده است. این قسمت ها برای راحتی کار و دیدگاه ماژولار تفکیک شده‌اند که به راحتی میتوان آن‌ها را توضیح داد و کتابخانه‌ها و منطق‌های مورد استفاده را و روش‌های پیاده‌سازی و مشکلات به وجود آمده

در حین انجام کار توضیح داده می‌شود.

در بخش چهارم بـه جمـع بنـدی و درستسـنجی سیسـتم و نحـوه‌ی اجـرای سیسـتم خـواهیم پرداخت. در این بخش نتایج به دست آمده به تفصیل قرار داده شده است.

در آخر در بخش پنجم به نتیجه‌گیری، پتانسیل موجود برای دستگاه مذکور و پیشنهادهایی برای بهبود سیستم برای شرکت‌های صنعتی قرار داده شده است.

فصل ۲

مفاهیم اولیه

در این فصل ابتدا مفاهیم اولیه را برای درک عمیق‌تر پروژه و تکنولوژی‌های استفاده شده همچون شبکه‌های عصبی، نحوه نمونسه برداری صدا و تبدیل به ویژگی‌ها، میکروکنترلرها و معماری‌های پردازشی مختلف را شرح میدهیم.

مقدمه

در فصل اول گریزی بـه سـاختار کلی سیسـتم کـه از ۲ بخش اصـلی نرم‌افـزاری و سـخت‌افزاری تشکیل شده است زده شد. در این بخش مفاهیم اولیه و تکنولوژی‌هایی که در هر بخش اسـتفاده شـده است شرح داده می‌شود. بخش نرم‌افزاری از یک شبکه عصـبی تحت فریمـوورک تنسـورفلو پیاده‌سـازی شده است. همچنین برای دریافت صوت و استخراج ویژگی از آن نیازمند الگوریتم‌هایی اسـت کـه طبـق آن بتوان یک ساختار کلی از صوت بدست آورد که خوراک شبکه عصبی شود. این الگـوریتم بـه صـورت کلی MFCC نام دارد. در ادامه نیازمندی‌های سخت‌افزاری و دنیای سیستم‌های نهفته و میکروکنترلر ها بررسی می‌شود که ساختارها و معماری های پردازشی و تجمعی آنها و همچنین دلایل اسـتفاده از آنهـا شرح داده می‌شود.

۲.۱- بخش اول: شبکه‌های عصبی

پیشرفت چشمگیر در عرصه پردازش‌های موازی و بـه روی کـار آمـدن کارت‌هـای گـرافیکی کـه توانایی انجام محاسبات ساده اما در عین حال در مقیاس بزرگ و همزمـان را دارنـد، بـاعث شـد کـه در زمینه یادگیری ماشین مخصوصاً یادگیری شـبکه‌های عصـبی و عمیـق پیشـرفتی بسـیار حاصـل شـود. یادگیری ماشین که با هـدف تصـمیم‌گیری و پیش‌بیـنی شـرایط توسـط ماشـین ایجـاد شـد، ابتـدا بـا یادگیری‌های آماری و گراف‌ها آغاز شد. این نوع یادگیری‌ها که سرچشمه در آمار و احتمالات داشتند به صورت کلی به صورت یادگیری با نظارت و بدون نظارت تقسیم بندی می‌شدند. سپس ایده‌ی استفاده از نوع عمل کرد مغز باعث شد که سیستمی به وجود بیاید که بتواند مانند مغـز یـادگیری را انجـام دهـد. پیشرفت سخت‌افزارها در دهه ۲۰ میلادی باعث شد که شبکه‌های عصبی رونق بسیاری بگیرند و بتوانند در جامعه یادگیری ماشین، خودی نشان دهند.

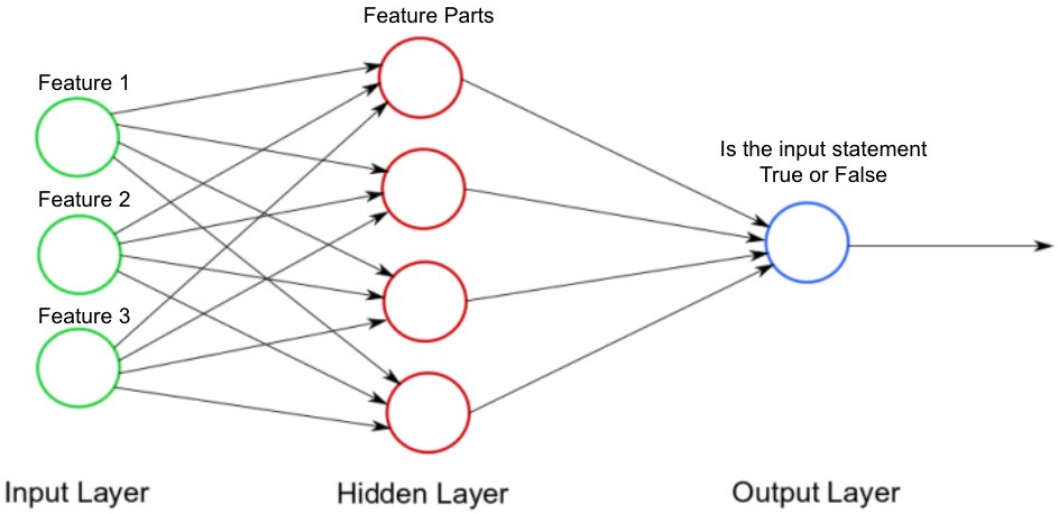
۲.۱.۱- توضیح شبکه‌های عصبی

شبکه‌های عصبی، سیستم ها و روش‌های محاسباتی نوین برای یادگیری ماشینی، نمـایش دانش و در انتها اعمال دانش بـه دسـت‌امده در جهت پیش‌بیـنی پاسـخ‌های خـورجی از سـامانه‌های پیچیـده هستند.[۵] ایده اصلی آنها الهام گرفته شده از سـاختار نـورنی زیسـتی ذهن انسـان اسـت. نورون‌هـا در شبکه‌های عصبی عناصر اصلی پردازشی فوق‌العاده به هم پیوسته هستند که به صورت هماهنـگ بـا هم عمل کرده و اطلاعات را به صورت سیناپس‌ها به یکدیگر منتقل می‌کنند.

نورون‌ها در شبکه‌های عصبی کوچک‌ترین واحد پردازشی و عملیات ریاضـی هسـتند. بـه صـورت کلی نورون‌ها در شبکه‌های مصنوعی عصبی، پردازش‌های ساده نظیر ضرب و جمع را انجام میدهند، امـا میتوانند محاسبات پیچیده‌تر مانند محاسبه توابع غیر خطی را نیز انجام دهند.

ساختار یک شبکه‌ی عصبی به صورت کلی از سه لایه تشکیل شده است. لایه‌ی ورودی که دریافت اطلاعات را محیط بیرون انجام میدهد. لایه‌ی میانی (یا به صورت کلی تر لایه‌های میانی که به لایه‌های پنهان معروف هستند) که با توجه به داده‌ها یادگیری اصلی شبکه را بر عهده دارند و لایه‌ی خروجی که با توجه به یادگیری لایه‌های پنهان، خروجی و نتیجه کلی شبکه را به محیط بیرون انتقال میدهند. نورون‌های هر لایه عموما با تمام نورون‌های لایه‌های مجاورش ارتباط دارند مگر آنکه قیدهایی از سمت طراح شبکه باعث محدودیت در ارتباط نورون‌ها با یکدیگر شده باشد. ارتباط بین نورون‌ها به صورت ضرایبی در نظر گرفته می‌شود که هدف اصلی شبکه‌ی عصبی یادگیری این ضرایب است.

به صورت کلی شبکه‌ی عصبی مجموعه‌ای از نورون‌ها میباشد که به محاسبات خطی یا غیر خطی انجام میدهند که در مجموع رفتار کلی شبکه، برآیند رفتار نورون‌های متعدد هست که خروجی را با استفاده از ورودی‌ها حاصل میکند.

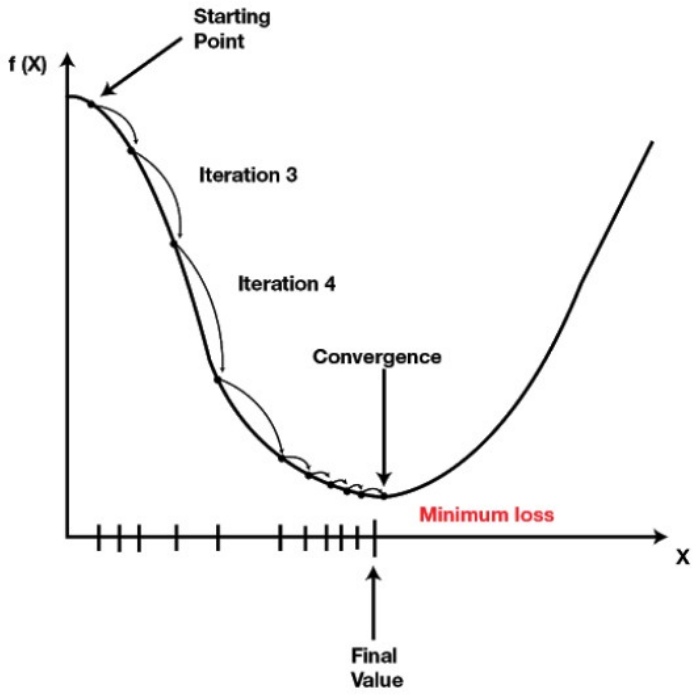


شکل (۱-۲) - ساختار کلی شبکه عصبی. لایه‌ی ورودی (Input Layer) که شامل ورودی ویژگی‌های لازم یادگیری است. لایه‌ی پنهان(Hidden Layer) که شامل لایه و یا لایه‌های میانی است که مسئول اصلی یادگیری هستند. لایه‌ی خروجی(Output Layer) که خروجی نهایی سیستم(در شکل درست بودن یا نبودن) را نشان میدهند و به دنیای خارجی، پیش‌بینی را اطلاع میدهند.

یادگیری در ماشین به ۳ دسته کلی تقسیم می‌شود. یادگیری با نظارت که به ازای ویژگی‌های موجود خروجی سیستم مشخص است. یادگیری بدون نظارت که داده‌ها و ویژگی‌ها که خروجی خاصی ندارند و به طور کلی گفته می‌شود که داده‌ها لیبل ندارند و دسته سوم که یادگیری تصویتی که ماشین به صورت یک عامل در محیط عمل میکند و با توجه به پاداش‌ها و مراحلی که در محیط دارد پیش رفته و یادگیری را انجام میدهد. در شبکه‌های عصبی عموما یادگیری به صورت با نظارت انجام می‌شود. در این نوع هدف ماشین این است که تابع هزینه‌ی خود را با توجه به داده‌ها بهینه کند. توابع هزینه انواع مختلفی دارد که در ساده‌ترین حالت به صورت اختلاف خروجی از مقدار واقعی است که هدف کمینه کردن آن است. در شبکه‌های عصبی مشکلی که وجود دارد این است که بهینه سازی تابع هزینه

به صورت محدب نیست. به این معنی که سیستم میتواند در کمینه‌های محلی قرار بگیرد و خروج از آن باعث مشکلاتی شود. برای حل این موضوع یکی از روش‌های متداول، استفاده از الگوریتم بازگشت به عقب[1] میباشد.

روش بازگشت به عقب به گرادیان تابع هزینه را برای تمام وزن‌های شبکه عصبی محاسبه میکند و سپس با روش‌هایی برای پیدا کردن مقدار بهینه مانند گرادیان کاهشی[2] به سمت وزن‌های بهینه اتصالات نورون‌ها حرکت میکند. این روش سعی دارد که در جهت خلاف گرادیان تابع (تغییرات تابع) حرکت کند به همین منظور لازم است که در شبکه‌ی عصبی از خروجی به عقب محاسبات انجام شود. به این صورت که گرادیان خروجی به صورت یک مشتق قابل محاسبه است اما در لایه‌های میانی لازم است که به صورت مشتق زنجیره‌ای عمل کرده و از انتها به لایه‌ها میانی و سپس به لایه‌های اولیه برسد و جهت حرکت تابع را مشخص کند.



شکل (۲-۲) - نحوه محاسبه گرادیان کاهشی. مقدار تابع از یک نقطه شروع با استفاده از توالی مختلف سعی دارد که در مخالف جهت افزایش تابع حرکت کند و به مقدار بهینه یا همگرایی برسد.

۲.۱.۲- انواع شبکه عصبی

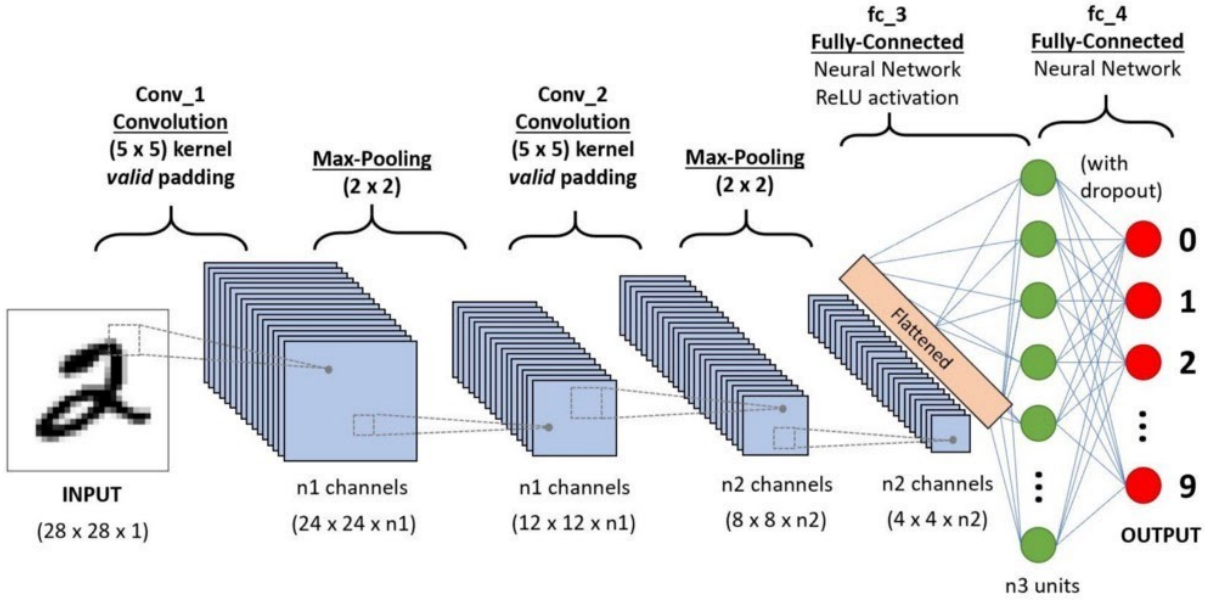
برای دسته بندی و انواع شبکه‌های عصبی روش‌های زیادی وجود دارد. یکی از روش‌های مرسوم که شبکه‌ها را دسته بندی میکنند بر مبنای لایه‌هایی که در شبکه استفاده می‌شود میباشد. لایه‌هایی

[1]Back Propagation
[2]Gradient Descent

که در شبکه‌های عصبی وجود دارد بر اساس نوع محاسبات و روش اتصال به همدیگر به سه صورت کلی شبکه‌های هوش مصنوعی[1]، کانوولشنال[2] و بازگشتی[3] تقسیم‌بندی می‌شود.

ANNs به صورت کلی شبکه‌هایی هستند که لایه‌های میانی عادی که مجموعی از نورون‌هاست دارند. معمولاً با اتصالات و تعداد نورون‌های این شبکه‌ها تغییرات را انجام میدهند که خروجی سیستم به صورت مطلوب بدست آید. شبکه‌های عصبی مخصوصاً در این نوع میتوانند عمیق یا کم عمق باشند. زمانیکه شبکه فقط یک لایه پنهان داشته باشد به آن کم عمق گفته می‌شود و بیشتر از یک لایه، عمیق خوانده می‌شود.

شبکه‌های کانوولشنال (CNNs) که با توجه به عملیات ریاضی کانوالو کردن دو سیگنال با هم به وجود آمده‌اند بیشتر برای شبکه‌هایی عصبیی که ورودی و ویژگی‌های داده‌های تصویری را میخواهند پردازش کنند به وجود آمده‌اند. این شبکه‌ها دارای لایه‌های کانوولشنال هستند که میتوانند ورودی لایه را با یک پنجره‌ای کانوالو کرده و خروجی را به لایه‌های بعدی دهند. این لایه‌ها معمولاً باعث ایجاد ویژگی‌های جدید در داده‌ها یا کشف ویژگی‌ها مانند مرزهای یک شکل استفاده شوند.

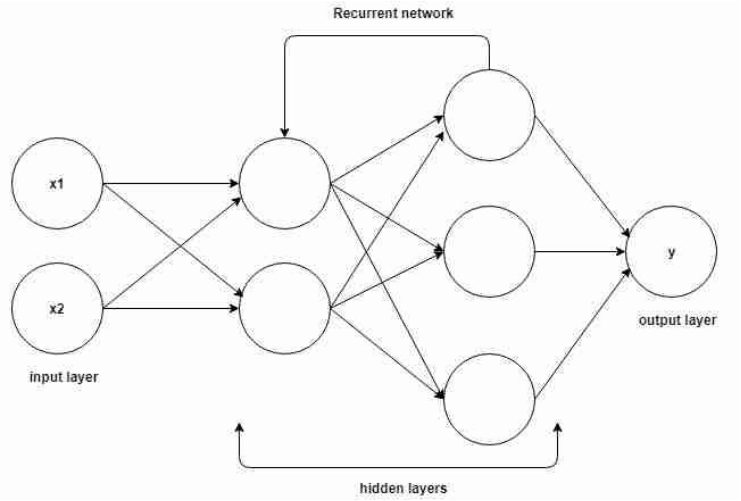


شکل (۳-۲) - ساختار یک شبکه‌ی CNN – در این شبکه، یک تصویر با ابعاد ۲۸*۲۸ دریافت شده و به عنوان خوراک به لایه‌های کانوولشنال داده می‌شود تا بتواند ویژگی‌های جدید یا کاهش ویژگی‌ها را انجام دهد. در انتهای شبکه یک لایه تمام متصل عادی قرار دارد. ساختار شبکه ۲ لایه کانوولشنال و یک لایه تمام متصل عادی قرار دارد.

شبکه‌های بازگشتی یا به اختصار RNNs برای داده‌های سری زمانی کاربر فرآوانی دارند. این شبکه‌ها برای پردازش داده‌های ترتیبی همچون صوت طراحی شده اند. این شبکه‌ها به ترتیب داده‌های ورودی حساس میباشند و دارای حافظه میباشند. در پردازش صوت و تبدیل گفتار در متن این نوع

[1] Artificial Neural Networks (ANN)
[2] Convolutional Neural Networks (CNN)
[3] Recurrent Neural Networks (RNN)

شبکه‌ها استفاده فراوانی دارند. ساختار کلی آن‌ها بسیار شبیه ANNs میباشد با این تفاوت کـه خـوراک لایه‌ها میتواند هم از لایه قبلی باشد هم از لایه بعدی و داده‌ها بعد از پردازش بـه صـورت بازگشـتی بـه لایه‌های قبلی داده می‌شوند.



شکل (۴-۲) - ساختار کلی یک شبکه بازگشتی. این شبکه در لایه اول پنهان خود، از لایه بعدی نیز تغذیه می‌شود.

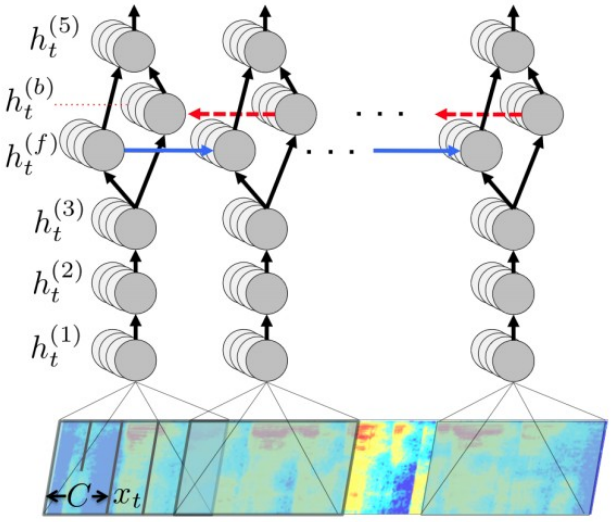
۲.۱.۳- استفاده شبکه‌های عصبی در صوت

صوت یک سیگنال زمانی است که به صـورت جریـانی پیوسـته در زمـان جریـان دارد. بـر خلاف سیگنال‌هایی همچون تصویر و یا حتی فیلم، نیاز است کـه صـوت بـه صـورت پیوسـته بررسـی شـود و داده‌های قبلی با حال رابطه دارند. به همین دلیل برای پردازش آنها نیاز است که از شـبکه‌های عصـبیی همچون RNN ها که به صورت بازگشتی عمل میکنند استفاده شود. در بررسی و تبدیل گفتار بـه متن، شبکه‌هایی مانند CNN و یا ANN نمیتوانند دقت بالایی داشته باشند.

در پیاده سازی صوت با شبکه‌های عصبی، مقاله‌ها و تحقیقات زیـادی انجـام شـده اسـت. یکی از پیاده‌سازی‌ها و ساختارها از شبکه‌های عادی ANNs و CNNs استفاده میکند. این پیاده سازی در یـک مقالـه تحت عنـوان "Building DNN Acoustic Models for Large Vocabulary Speech Recognition" می‌باشد که با استفاده از ۲۱۰۰ ساعت داده شبکه‌ی خود را آمـوزش داده‌انـد و بـه دقت حدود ۶۰ درصد رسیده‌اند.

همانطور که در مقدمه گفته شد، در سال ۲۰۱۴ نیز شرکت بایـدو، سـاختار کلـیی بـرای تبـدیل گفتار به متن توسط شبکه‌های عصـبی بازگشـتی ارائـه کـرد. مـزیت این شـبکه اسـتفاده از شـبکه‌های

بازگشتی، تیم بسیار بزرگ این شرکت و پیاده سازی آن توسط خودشان بر روی مجموعه داده‌های معروف همچون Switchboard و WSJ پیاده سازی و تست شده است. خطای این سیستم با توجه به گزارشی که در مقاله خود داده‌اند در حدود ۷ درصد بوده است. ساختار کلی این شبکه به صورت ۵ لایه پنهان میباشد که ۲ لایه آن به صورت بازگشتی با هم تعامل دارند. [۲]



شکل (۵-۲) - ساختار شبکه‌ی پیاده‌سازی شده توسط شرکت بایدو

۲.۲- بخش دوم: الگوریتم‌های نمونه برداری از صوت و استخراج ویژگی

سیگنال‌های صوتی، به صورت سری زمانی میباشند. به این صورت که در طول زمان موج صوتی حرکت میکند و در ذرات هوا ارتعاش ایجاد میکنند. نمونه برداری از این سیگنال برای سیستم‌های ماشینی یکی از بحث‌های جالب و سخت هست. در این بخش به توضیح الگوریتم‌های مرسوم نمونه برداری از صدا و دلایل استفاده یا عدم استفاده از آن‌ها میپردازیم

۲.۲.۱- سیگنال صوتی

انسان‌ها احساسات، عقاید و نظرات خودشان را با استفاده از صدا و صحبت به یکدیگر منتقل میکند. روند تولید صحبت همراه گفتار، صدا و تسلط است. این مورد یکی از توانایی‌های پیچیده انسان است که او را از دیگر جانوران متمایز میسازد. تارهای صوتی انسان در هر ثانیه میتوانند بیش از ۱۴ صدای مختلف را با استفاده از ۱۰۰ ماهیچه با فرمان مغز و ستون فقرات تولید کنند.
توسط محققان، چندین تلاش موفق برای تولید و ایجاد یک سیستم بررسی، دسته‌بندی و

شناسایی سیگنال‌های صوتی و صحبت انجام شده است. تلاش‌ها هم در سمت سخت‌افزار و هم نرم‌افـزار در حوزه‌های فراوانی همچون پزشکی، بخش‌های دولتی وکشاورزی میباشد.

سیگنال‌های صوتی توسط سنسورهای دیجیتال ماننـد میکـروفن و یـا Phonograph cartridge دریافت می‌شوند. خروجی الکتریکی مبدل یک سیگنال آنالوگ نامیده می‌شود زیـرا سـیگنال الکـتریکی مشابه الگوری فشار موج صوتی است که آن را بوجـود آورده اسـت. سـیگنال‌های صـوتی بـه صـورت دو بعدی در این گیرنده‌ها ذخیره می‌شود که محور عمودی شـدت سـیگنال و محـور افقی زمـان را نشـان میدهد خود این سنسورها نیز به این صورت عمل میکنند که از سیگنال اصـلی آنـالوگ محیطی نمونـه برداری می‌کنند که یک نرخ نمونه برداریی دارد. هر چه نرخ بیشـتر باشـد، از داده‌هـای موجـود بیشـتر نمونه برداری شده و صدای دریافت شده کیفیت و اطلاعات بیشتری خواهد داشت.

۲.۲.۲- مشکلات استخراج ویژگی

سیگنال صوتی، در اکثر مواقع همراه نویز محیطی و نویز دستگاهی بدلیل کمبود نمونه بـرداری و مشکلات می‌باشد. به همین دلیل در ابتدا باید صوت عاری از نویز شـود کـه بتـوان بعـد از آن از صـوت ویژگی‌های لازم را استخراج کرد. هدف استخراج ویژگی از صوت روشن سازی سیگنال و در ادامه تحلیل بخش‌های مختلف آن است. دلیل آن این است که در سیگنال صـوتی اطلاعـات زیـادی وجـود دارد کـه برای کارشناسی و بررسی آن حجم محاسبات سنگینی داریم.

استخراج ویژگی از سیگنال صوتی از تبدیل موج صوتی به این صورت میباشد که بـا پارامترهـایی که در نرخ داده نسبتاً کمتری دارند اما اطلاعات بیشتری در رابطه به مـوج و تحلیل‌هـای بعـدی بـه مـا میدهد انجام می‌شود.

سیگنال صوتی ذخیره شده در سیستم‌های دیجیتال، به صورت آرایه‌ای از زمان و شدت می‌باشد. این سیگنال‌ها در حوزه زمان ذخیره سازی شده‌اند و اطلاعاتی که دارند از جمله شدت و فرکانس صـدا، در این حوزه میباشد. برای تحلیل سیگنال‌های زمانی و استخراج اطلاعات از آن‌ها مشکلاتی وجود دارد. برای مثال اطلاعات زمانی صدا و شدت به خودی خود ویژگی‌هایی که بتوان از آن اطلاعاتی استخراج یـا استفاده کرد ندارند. به همین دلیل استخراج ویژگی‌ها برای دسته بندی اطلاعات، امری ضروری است.

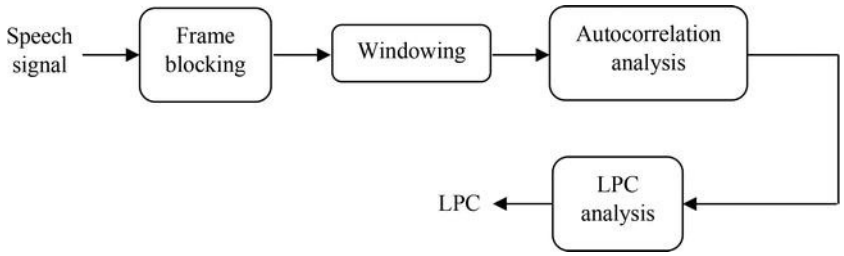
۲.۲.۳- انواع الگوریتم‌ها

برای استخراج ویـژگی از سـیگنال‌های صـوتی الگوریتم‌هـای مختلفی وجـود دارنـد. معمـولاً این الگوریتم‌ها یک بردار چند بعدی از ویژگی‌های صـوت را همـراه خـود دارنـد. بـرای نمـایش پارامترهـا و استخراج ویژگی رویکردهای مختلفی وجود دارد.

پیش‌بینی خطی ادراکی[1] یکی از روش‌های استخراج ویژگی است که ضرایب ثـابتی از روی طیـف پوش سیگنال[2] بدست میاید. این ضرایب از اندازه لگاریتم طیف فرکانسی سـیگنال کـه از تبـدیل فوریـه

[1] perceptual linear prediction
[2] Spectral Envelope

بدست آمده‌اند محاسبه می‌شوند. این روش بیشتر برای پردازش کلی سیگنال با توجه به اطلاعات محدود جمع آوری شده میباشند. این الگوریتم نسبت به نویز کمتر آسیب‌پذیر است چون میتوان از طیف خروجی توان‌هایی که فرکانس بالایی دارند حذف کرد.



شکل (۶-۲) - بلاک دیاگرام پردازش به روش LPCC

یکی از روش‌های دیگر که بر اساس شنوایی انسان طراحی شده است به نام Mel frequency cepstral coefficients (MFCC) که برای مشخص کردن و شناسایی کلمات داخل جمله به کار میرود. این الگوریتم با استفاده از پهنای باند گوش انسان که میتواند بازه‌ی خاصی را شناسایی کند طراحی شده است. در بخش بعدی به تفضیل به بررسی این الگوریتم میپردازیم.

الگوریتم های دیگری نظیر LPCC, PLP, LSF و ... وجود دارد که از حوصله این بحث خارج است.

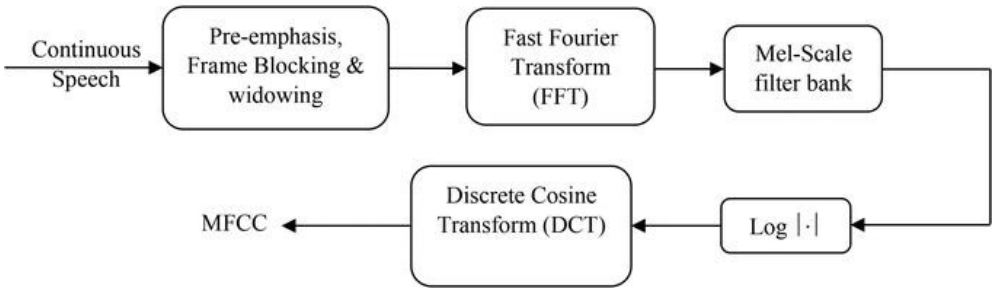
۲.۲.۴- الگوریتم MFCC

ضرایب کپسترال فرکانس-مل بهترین و شناخته شده‌ترین ویژگی گفتار میباشد که به طور گسترده برای دو هدف تشخیص گفتار و گوینده استفاده می‌شود. یک مل، یک واحد اندازه‌گیری مبتنی بر فرکانس درک شده گوش انسان است. مقیاس مل تقریباً فواصل فرکاسنی زیر ۱۰۰۰ هرتز و فواصل لگاریتمی بالای ۱۰۰۰ هرتز دارد.. تقریبی از مل برای فرکانس میتوان به صورت زیر بیان شود[۱]:

$$mel(f) = 2595 \log_{10}\left(1 + \frac{f}{700}\right)$$

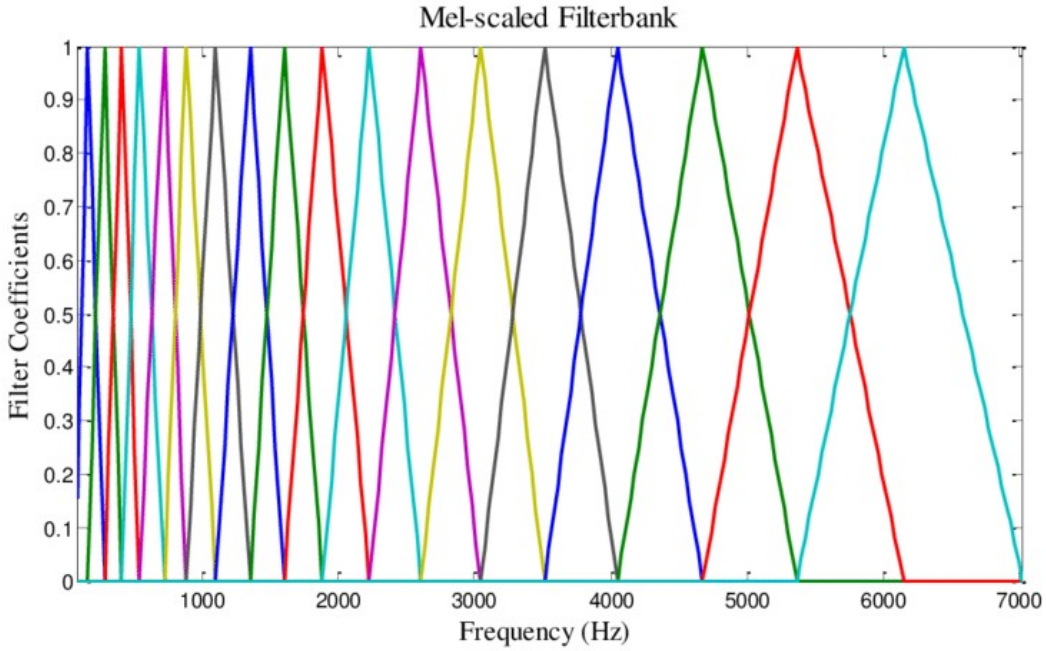
که در آن f به فرکانس واقعی و mal(f) به فرکانس درک شده اشاره دارد. باید توجه داشت که مهم‌ترین مزیت اصلی MFCC استحکام و مقاومت به نویز و خطاهای تخمین طیف تحت شرایط مختلف می باشد. با توجه به شکل زیر به منظور استخراج MFCC باید مراحل زیر صورت بگیرد:

۱. دریافت اطلاعات انالوگ و تبدیل به دیجیتال
۲. فریم بندی و پنجره کردن داده‌ها(به عنوان مثال هر ۱ ثانیه یک پنجره)
۳. تبدیل فوریه
۴. اعمال پردازش‌های لازم برای MFCC



شکل (۷-۲) - بلاگ دیاگرام تولید ضرایب MFCC

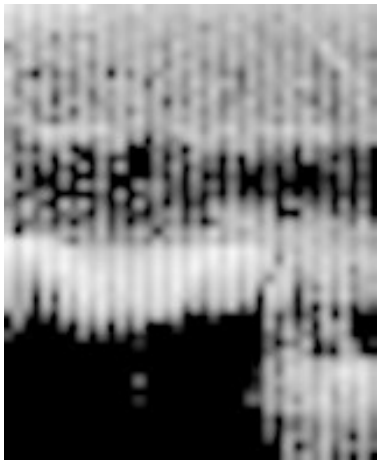
با توجه به شکل بالا، بعد از نمونه‌برداری مقداری تقویت می‌شود و به صورت فریم‌هایی در واحد زمان بدست میاید. نرخ‌های نمونه برداری عموما بر اساس فرکانس صدای انسان در حدود ۴۸ هزار بیت در ثانیه میباشد. از آنجایی که تغییرات در مجرای صوتی و صدای تولید شده یک فرایند غیر ایستان نمی‌باشد، با استفاده از فریم‌های ۳۰ میلی ثانیه صورت را تقسیم‌بندی میکنیم که سیگنال به شکل یک سیگنال شبه ایستان شود. بعد از تقسیم بندی سیگنال به قاب ها و فریم‌های ۳۰ میلی ثانیه‌ای به منظور جلوگیری از از دست رفع اطلاعات به دلیل برش سیگنال به اندازه‌ی ۳۰ تا ۵۰ درصد قاب‌ها را همپوشان در نظر میگیریم. سپس به منظور بدست آوردن فرکانس‌ها و محاسبه فرکانس مل از هر قاب تبدیل فوریه گرفته می‌شود. حال بعد از بدست آوردن فرکانس‌های مل، از یک فیلتر به شکل زیر اطلاعات عبور داده می‌شود تا سیستم مانند گوش انسان عمل کند.



شکل (۸-۲) - فیتر مل در فرکانس‌های مختلف

در نهایت، با تبدیل لگاریم طیف مل، سیگنال به حوزه زمان بر گشت داده می‌شود. نتیجه حاصل،

ضرایت کپسترول فرکانس مل MFCC نامیده می‌شود. از آنجا که ظریف طیفی مل و لگـاریتم آن اعـداد حقیقی هستند، میتوان با استفاده از بتدیل گسسته کسینوسی میتوان دوباره سیگنال را به حوزه زمـان بازگردانی کرد. همچنین میتوان از تبدیل را انجام نداد و با استفاده از ضرایب بدست آمـده یـک تصـویر طیف‌سنج ایجاد کرد. این تصویر که به spectrogram معروف است مانند اثر انگشت اون فـریم صـدا در نظر گرفته می‌شود. این تصویر را میتوان به عنوان یک تصویر به یک شبکه‌ی عصبی داد و آنـرا پـردازش کند و پترن‌های موجود را شناسایی کند. نمونه‌ای از spectrogram به شکل زیر است:



شکل (۹-۲)- نمونه‌ای از spectrogram برای یک ثانیه صوت کلمه yes

۲.۳- سیستم‌های نهفته (Embedded Systems)

۲.۳.۱- توضیح سیستم‌های نهفته

سیستم‌های نهفته یا سامانه‌های نهفته، سیتسم‌های کامپیوتری هستند که برای یـک هـدف و یـا عمل خاصی در یک سیستم مکانیکی یا الکتریکی بزرگ‌تـر قـرار میگیرنـد و اقلب بـه صـورت بیـدرنگ[1] پردازش‌های لازم را انجام میدهند.

سامانه های نهفته در طیف گسترده ای از وسایل، شامل ساعت های دیجیتـالی و MP3 Player ها، سیستم های بزرگ و ایستا مانند چراغ های راهنمایی، کنترلر های کارخانه ها و سیستم های بسـیار پیچیده مانند اتومبیل های هیبریدی، MRI و الکترونیک هوانوردی مورد استفاده قرار می گیرند. میزان پیچیدگی از مقداری بسیار کم، با یک چیپ میکروکنترلر تـا بسـیار پیچیـده بـا چنـدین واحـد نهفتـه، دستگاه های جانبی و شبکه هایی که در داخل یک شاسی یا محفظه بزرگ سوار می شـوند، تغیـیر می

[1]Real Time Processing

کند.

۲.۳.۲- دلایل استفاده

سیستم‌های بی‌درنگ کاربرها و استفاده‌های فراوانی دارند. مصرف توان کمتر، اندازه کوچک‌تر، بازه عملیاتی انعطاف پذیر و هزینه کمتربر به ازای هر واحد تولید باعث ایجاد ویژگی‌هایی می‌شود که به قیمت منابع پردازش محدود این سیستم‌ها بدست میاید که باعث می‌شود برنامه ریزی و تعامل با آنها دشوار باشد. با این وجود، ویژگی‌های زیاد آنها باعث می‌شود که در صنعت دشواری تعامل با آنها پذیرفته شود و از آنها در سیستم‌های بسیاری کاربرد داشته باشند. در چند سال اخیر با مکانیزم‌های هوش مصنوعی بر روی سخت‌افزار، بهره بردن از سنسورهای موجود و وجود یک شبکه از سامانه‌های نهفته یتوان منابع موجود در سطوع واحد و یا شبکه را بهینه کرد و قابلیت‌های زیادی را به آنها افزود.

به دلیل این که وظایف مشخصی به سامانه نهفته اختصاص داده می شود، مهندسین طراح می‌توانند آن را بهینه سازی کنند تا ابعاد و هزینه دستگاه کاهش یابد و عملکرد و اطمینان پذیری آن افزایش یابد. برخی سامانه های نهفته به صورت عمده تولید می شوند و از مزیت مقیاس پذیری بهره می‌برند.

۲.۳.۳- اجزا

سیستم‌های نهفته دارای هسته های پردازشی هستند که می توانند هر پردازنده RISC یا CISC نظیر میکروکنترلر(µC) ، میکروپروسسور (µP) یا پردازنده سیگنال‌های دیجیتال (DSP) باشند. پردازنده یک واحد مهم در سخت‌افزار یک سامانه‌ی نهفته است. مشخصه‌ی کلیدی این سیستم‌ها، طراحی اختصاصی برای انجام یک کار مشخص است. به این دلیل که سیستم های نهفته برای یک کار مشخص اختصاص یافته‌اند، مهندسین طراح می‌توانند محصول را برای کاهش اندازه و قیمت بهینه کرده و اطمینان پذیری و کارایی آن را بالا ببرند. برخی از سیستم‌های نهفته با بهره‌گیری از مزیت‌های تولید با تعداد بالا، به شکل انبوه تولید شده‌اند.

در یک سیستم نهفته کامل تر میتوان از یک FPGA در کنار یک CPU (میکروکنترلر یا میکروپرسسور) استفاده کرد. اما بعد از پردازنده که مهمترین بخش سیستم های نهفته می باشد ، نرم افزار ، سنسور ها ، مبدل های آنالوگ به دیجیتال ، مبدل های دیجیتال به آنالوگ ، فعال کننده ها و ... استفاده کرد که در شکل زیر مشاهده می کنید.

۲.۴- سخت‌افزار مورد استفاده

۲.۴.۱- توضیح کلی

نام میکروکنترلر از دو عبارت میکرو و کنترلر تشکیل یافته است. اولین معنایی که به ذهن‌ها می‌رسد کاربرد میکروکنترلر به عنوان یک کنترل کننده کوچک می‌باشد. در حقیقت این قطعات مانند یک کامپیوتر بسیار کوچک توانایی پردازشی دارند و به کمک ورودی/خروجی های مختلف می توانند با قسمت های دیگر مدار ارتباط برقرار کنند.

برای مثال زمانی را فرض کنید که شما می خواهید سرعت یک موتور با زدن یکی از کلیدها بیشتر شده و با زدن کلید دیگری کمتر شود. علاوه بر این یک کلید دیگر برای اینکه موتور با هر بار زدن کلید روشن و خاموش شود نیاز دارید. اینجا میکروکنترلر به کمک شما می آید و شما پردازش مورد نیاز برای این کار را توسط میکروکنترلر می توانید انجام دهید.

۲.۴.۲- برد STM32F746

بردهای دیسکاوری که متعلق به شرکت STM میباشد، جزو بردهای آموزشی و متناسب برای شروع به کار با میکروکنترلرهای این شرکت محسول میگردد.

برد دیسکاوری STM32F7 که دارای میکروکنترلر ۳۲بیتی بر مبنای پردازنده ARM سری Cortex®-M7 میباشد، به کاربران امکان نوشتن و به اشتراک گذاشتن برنامه هایشان را میدهد.

با کمک این برد میتوان رنج وسیعی از برنامه های کاربردی نظیر صدا، پشتیبانی از چند سنسور، گرافیک، امنیت، ویدئو و پورت های پرسرعت را توسعه داد.

همچنین با پشتیبانی از سوکت توسعه مبتنی بر Arduino میتوان رنج وسیعی از بردها و ماژول های دیگر را به آن متصل کرد.

همچنین این برد دارای یک نمایشگر LCD از نوع TFT به ابعاد ۴.۳ اینچ میباشد.

در این پروژه با توجه به نیازی که به میکروفن و نمایشگر داریم از این برد استفاده می‌کنیم. این برد از پشتیبانی خوب شرکت ST بهره میبرد و میکروکنترلر آن از سیستم قدرتمند Mbed نیز پشتیبانی میکند. مشخصات فنی آن در جدول زیر قابل مشاهده است:

جدول (۱-۲) - مشصخات میکروکنترلر استفاده شده

ویژگی	مقدار

فرکانس کاری	216 Mhz
میزان رم	128 Mb
میزان فلش	512 KB
کاکتورهای سریال	UART, SPI, I2C, USB, CAN
Timer	16bit – 32bit
Voltage	1.7 to 3.6 V

۲.۴.۳- فریمورک Mbed

یکی از مشکلات اساسی میکروکنترلرها برنامه نویسی سخت آنهاست. برنامه‌نویسی در سطح رجیستری در میکروکنترلرهای امروزی بدلیل افزایش ورودی-خروجی‌ها یکی از طاقت‌فرساترین کارها شده است. در این میان شرکت‌ها و گروه‌های مختلفی سعی کرده‌اند که یک پلفورم کلی برای اکثر میکروکنترلرهای شرکت‌های متخلف ایجاد کنند. از زاویه‌ای دیگر، سیستم عامل برای میکروکنترلرها بدلیل سرعت و حجم حافظه پایین آنها نمیتوان استفاده کرد. به همین منظور نیاز دیده شد که یک فریومور میانی که حداقل‌های سیستم عامل را دارا باشد و از سمتی به اندازه کافی سبک باشد که بتوان بر روی میکروکنترلرها استفاده کرد.

یکی از این فریمورک‌ها که با پشتیبانی شرکت STM شروع به ایجاد شدن کرد، فریمورک Mbed می‌باشد. این فریمورک که با هدف پشتیبانی اکثر میکروکنترلرهای معروف و آسانی کار برنامه نویس، ساختار پیدا کرده است.

Mbed به صورت کلی یک سیستم عامل با تعاریف آن می‌باشد که دامنه وسیعی از میکرکنترلرها و پروسسورهای مبتنی بر ARM را پشتیبانی می‌کند با استفاده از Mbed به راحتی میتوان سیستم‌هایی در سطح برای IOT ایجاد کرد. این سیستم عامل اکثر دیگر فریمورک‌ها مانند CMSIS-RTOS را پشتیبانی می‌کند و بسیاری از ویژگی‌های آنها را در خود جای داده است. همچنین این پلتفرم قابلیت MultiThread به کاربر میدهد که می‌تواند کارهای خود را برنامه ریزی کند و به صورت یک سیستم قدرتمند با آن برخورد کند. در این پروژه نیز با توجه به روند روز دنیا از این پلتفرم بهره برده شده است.

این پروژه با استفاده از پلتفرم Mbed پیاده‌سازی شده است. بدلیل استفاده زیاد از بخش‌های مختلف میکروکنترلر مانند تایمر، میکروفن و LCD و همچنین نیاز کتابخانه‌های Tensorflow که برای این سیستم‌عامل نوشته شده است صلاح دیده شد که از این پلتفورم استفاده شود.

۲.۵- خلاصه و جمع‌بندی

در این فصل نیازمندی‌های پیاده سازی پروژه بررسی شد به گونه‌ای که مفاهیم اولیه آشنا شـــویم تا بتوانیم در فصل بعد به پیاده سازی سیستم و توضیحات آن بپردازیم

فصل ۳

شبیه‌سازی و طراحی

در این فصل به نحوه‌ی آماده سازی محیط شبیه‌سازی و پیاده‌سازی سیستم می‌پردازیم.

مقدمه

سیستم تشخیص کلمات با استفاده از میکروکنترلر از ۲ بخش کلی تشکیل شده است. بخش اول مدل یادگرفته شده بر روی یک کامپیوتر قدرتمند است و تبدیل مدل یادگرفته شده به یک مدل بسیار سبک و بخش دوم آن پیاده سازی بر روی میکروکنترلر که یک کامپیوتر بسیار ساده و ضعیف میباشد.

برای پیاده سازی مدل، ابتدا باید یک مجموعه داده جمع آوری کرد. سپس با طراحی شبکه بر روی شبکه‌های عصبی و آموزش شبکه با استفاده از داده، یک شبکه‌ی عصبی که قابلیت تشخیص کلمات مورد نظر را دارد بدست میاید. این شبکه باید در این مرحله به بیشترین میزان دقت خروجی برسد. به همین منظور نیاز از الگوریتم‌های مختلف برای افزایش دقت شبکه استفاده کرد.

قسمت سخت افزاری سیستم، با استفاده از یک میکروکنترلر پیاده سازی می‌شود. در این مرحله با استفاده درگاه‌های ورودی و خروجی، میکروکنترلر می‌تواند داده‌های محیطی صوتی را دریافت کند و در حافظه خود ذخیره کند. سپس یک ماژول داده های تحلیل کرده و به صورت طیف‌سنج (spectrogram) درآورده و به شبکه عصبی میدهد. سپس خروجی شبکه عصبی با استفاده بررسی شده و خروجی کلی سیستم به کاربر نشان داده می‌شود.

۳.۱- آموزش مدل و ساختار آن

برنامه نوشته شده بسیار عمومی به صورتی که میتوان با کلمات دیگری با تغییرات بسیار جزئی استفاده کرد. در این بخش ابتدا در مورد مجموعه داده‌ای که استفاده شده است صحبت می‌شود و در ادامه به آماده سازی محیط و نحوه‌ی آموزش مدل و ساختار آن و در انتها به نحوه‌ی خروجی گرفتن مدل اختصاص داده شده است.

۳.۱.۱- داده‌های مورد استفاده

در این پروژه با توجه به نیاز زیاد به داده از مجموعه داده‌های جمع آوری شده توسط گوگل استفاده می‌شود. این مجموعه داده که شامل ۳۰ کلمه اصلی می‌باشد، بیش از ۱۰۰ هزار صدای یک ثانیه‌ای در خود جای داده است. این مجموعه داده توسط شرکت گوگل تحت پروژه متن باز ارائه می‌شود.

این مجموعه داده به صورت انگلیسی جمع آوری شده است که کلماتی مانند yes, no, up, down, left, right, stop, ... می‌باشد. سیستم مورد استفاده در این بخش با توجه به ساختار استفاده شده در این مجموعه داده که به صورت فولدر بندی دانلود می‌شود ساختار یافته است. این مجموعه داده کنار داده‌های مورد نظر دارای یکسری مجموعه داده به عنوان نویز محیطی مانند صدای خیابان و بوق ماشین نیز هست. برای دستیابی به مجموعه داده میتوان از لینک زیر استفاده کرد:

https://aiyprojects.withgoogle.com/open_speech_recording

۳.۱.۲- آماده سازی محیط برای آموزش مدل

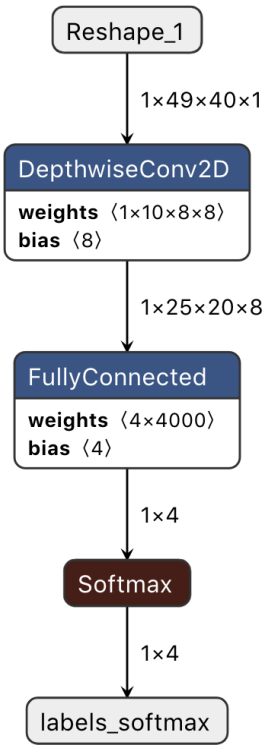
برای آماده سازی محیط جهت آموزش مدل نیاز است که نرم‌افزارهایی بر روی سیستم نصب باشد. برای آمده سازی محیط ابتدا لازم است که زبان برنامه نویسی پایتون بر روی سیستم نصب باشد. مدل آموزش داده شده در این پروژه از پایتون ۳.۶ استفاده شده است. بعد از نصب پایتون متناسب با سیستم عامل مورد استفاده نیاز است که کتابخانه‌های مورد نیاز برای اجرای فایل‌های اجرایی نصب گردد. در این پروژه با استفاده از ویژگی‌های پایتون فایلی با عنوان requirements.txt در کنار پروژه قرار دارد که با استفاده از دستور زیر میتوان به راحتی کتابخانه‌های مورد استفاده را نصب کرد. برای اجرا لازم است که محیط ترمینال سیستم عامل که پایتون بر روی آن نصب است به آدرس پروژه برده و دستور را اجرا کنید:

```
pip install -r requirements.txt
```

این دستور ممکن است که کمی زمان ببرد. دلیل آن این است که کتابخانه tensorflow از حجم بالایی برخوردار است بدلیل سرعت اینترنت پایین، دانلود آن و نصب آن زمانی را طول میکشد.

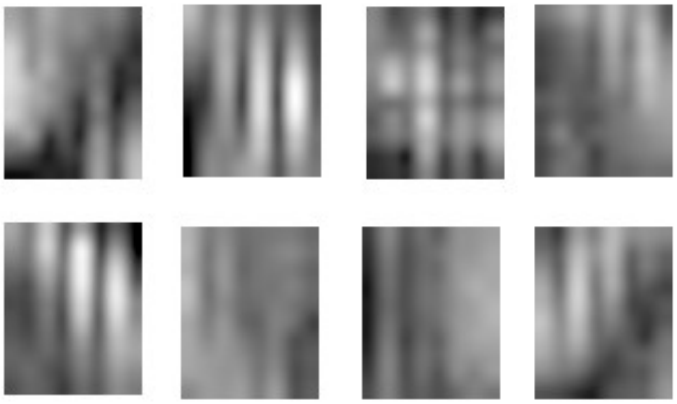
۳.۱.۳- ساختار شبکه عصبی

در این قسمت به ساختار کلی شبکه عصبی می‌پردازیم. شبکه عصبی استفاده در این پروژه با توجه به قیودی که داریم یک شبکه گرافی کوچک می‌باشد. این مدل شامل یک لایه Convolutional همراه با یک لایه Fully Connected و در ادامه یک لایه Softmax برای فعال سازی خروجی ها میباشد. در شکل زیر لایه‌ی convolutional با عنوان DepthwiseConv2D قرار داده شده است.



شکل (۱-۳) - ساختار کلی شبکه مورد استفاده

لایه‌ی Convolutional برای بدست آوردن ویژگی‌های تصویر ورودی مورد استفاده قرار میگیرد. ۸ فیلتر مورد استفاده در این لایه ارایه‌های مستطیلی می‌باشند که به صورت پنجره‌ای بر روی تصویر حرکت می‌کنند. این فیلترهای ۸ در ۱۰ میزان شباهت پیکسل‌های مجاور به یکدیگر را نشان می‌دهند. این فیلترها توسط تیم tensorflow از با استفاده از مقاله‌ای که دانشگاه استنفورد عمومی کرده است استخراج شده است[۳]



شکل (۲-۳) - فیلترهای استفاده شده در لایه‌ی اول

هر فیلتر مانند یک تیکه کوچک از ورودی ها می‌باشد که شبکه سعی دارد که پترن مورد نظر را پیدا کند. هر چه شباهت بیشتر باشد، مقدار خروجی فیلتر بیشتر می‌شود و در تصویر خروجی مقدار آن بیشتر ثبت می‌شود.

چون در این ساختار ۸ فیلتر وجود دارد، خروجی لایه‌ی اول یک تصویر ۸ لایه‌ای می‌باشد.برای ورود به لایه‌ی بعدی یک کاهش ابعاد بر روی تصویر انجام می‌دهیم که پیکسل‌ها را یکی در میان انتخاب میکنیم . به همین دلیل انگار تصویر خروجی نصف تصویر ورودی خواهد بود. ورودی لایه‌ی بعدی به صورت کلی مانند یک تصویر برای لایه‌ی بعدی می‌باشد.

لایه‌ی بعدی شبکه یک لایه تمام متصل است. عملکرد این لایه با لایه‌ی قبلی متفاوت است. برخلاف لایه قبل که بر روی هر برش تصویر ورودی یک عملیات انجام می‌شود، در این لایه به ازای هر ورودی یک وزنی وجود دارد. هدف این است که بر اساس ورودی‌های مشخص، بهترین وزنی که باعث شود خروجی بدست آید محاسبه و تخمین زده شود. این بخش مانند پیدا کردن یک الگوی سراسری در داده‌ها به زمانیکه به ازای یک ورودی انتظار داریم خروجی دیده شود.

لایه‌ی آخر softmax می‌باشد. این لایه به طور موثر کمک به جدا سازی بین بالاترین عدد خروجی و تفاوت بین ورودی‌ها می‌شود که باعث نمی‌شود که ترتیب آنها عوض شود. خروجی این لایه معمولا به صورت احتمالی در نظر گرفته می‌شود که بسیار برای انتخاب امتیاز مناسب است.

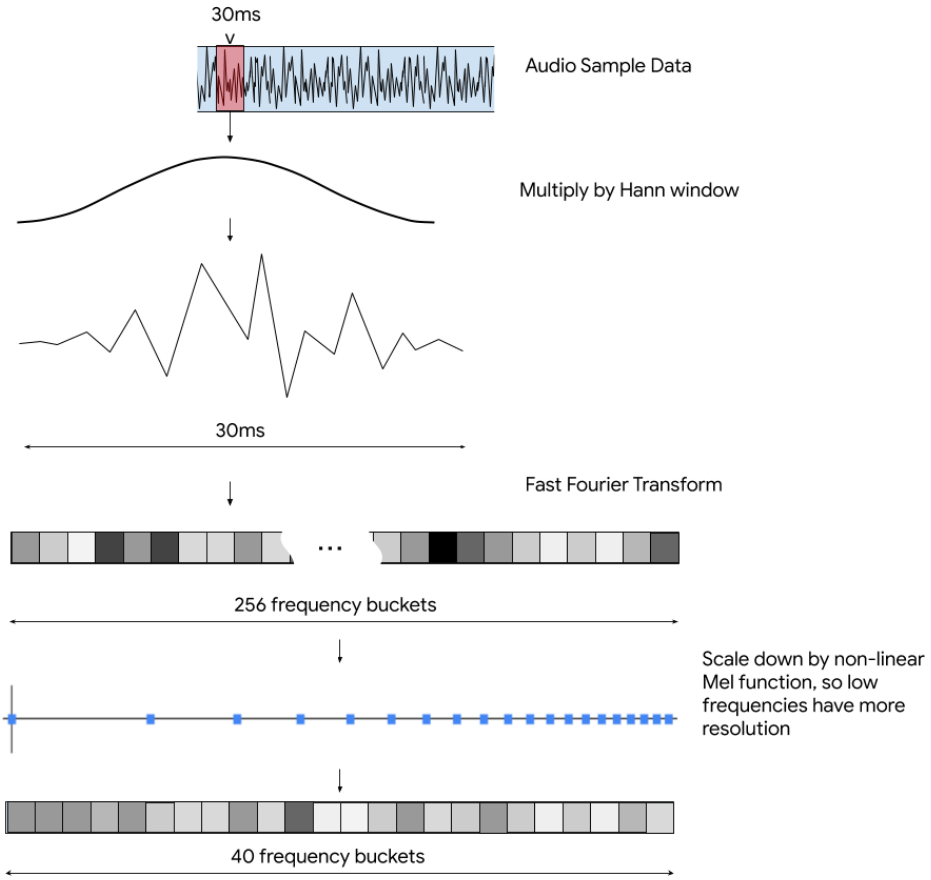
جدای از لایه‌های اصلی سیستم برای همگرایی بیشتر سیستم و افزایش سرعت آن، وزن‌های به صورت کاتوره‌ای در لایه‌های دو لایه اصلی قرار داده شده‌اند و برای تابع فعال سازی نیز از تابع ReLU استفاده شده است که این تابع باعث می‌شود که اعداد کمتر از صفر در خروجی پدیدار نشود.

برای بدست آوردن مدل شبکه، با توجه به محدودیت فضای ذخیره سازی بر روی میکروکنترلر و دقت‌های لازم، ساختارهای مختلفی بررسی شد. در فصل بعد تعدادی از ساختارها و نتایج آنها و علل کنار گذاشتن انها بررسی می‌شود.

۳.۱.۴- استخراج ویژگی‌ها

همانطور که در بخش قبلی در مورد استخراج ویژگی از سیگنال صوتی توضیح داده شد، در این پروژه از الگوریتم MFCC استفاده شده است. مراحل استخراج ویژگی‌ها از تبدیل فوریه شروع می‌شود. صوت به صورت برش‌های ۳۰ میلی ثانیه تقسیم بندی می‌شود و از فیلتر زنگوله‌ای برای حذف نویزها استفاده می‌شود. تبدیل فوریه یک عدد مختلط به ازای هر فرکانس تولید می‌کند اما تمام چیزی که برای ما ارزش دارد، انرژی کلی آن است که از اندازه‌ی اعداد می‌باشد.

به ازای N نمونه داده شده، تبدیل فوریه نصف آنها فرکانس تولید می‌کند با توجه به نرخ نمونه برداری ۱۶۰۰۰ نمونه در ثانیه، ۳۰ میلی‌ثانیه داده، ۴۸۰ نمونه می‌شود که به صورت کلی ۲۴۰ فرکانس تولید می‌کند. با توجه به اینکه الگوریتم‌های FFT نیاز دارند که توانی از ۲ نمونه دریافت کنند، به همین دلیل ما به تبدیل فوریه ۵۱۲ نمونه میدهیم که به ما یک مجموعه ۲۵۶ تایی فرکانس می‌دهد. این تعداد بیشتر از نیاز ما می‌باشد به همین دلیل ما به ۴۰ نمونه آن را کاهش می‌دهیم این کاهش نمونه به صورت خطی نمی‌باشد و بر اساس گوش انسان این کاهش نمونه صورت می‌گیرد که فرکانس‌هایی که گوش به آنها حساس نیست را حذف میکند. به صورت کلی شکل زیر ساختار ایجاد بسته‌های ۴۰ تایی فرکانس را نشان میدهد:



شکل (۳-۳) - نحوه‌ی پردازش سیگنال و اسخراج دسته‌های فرکانس

این عملیات ۴۹ بار دیگر برای صوت‌های یک ثانیه‌ای که صورت برش‌های ۳۰ میلی ثانیه‌ای تقسیم شده‌اند انجام می‌شود. برای جلوگیری از از دست رفت داده بدلیل برش، پنجره‌ای که بر روی ۱ ثانیه صوت حرکت می‌کند، ۱۰ میلی ثانیه با قبلی همپوشانی دارد پس پنجره ۲۰ میلی ثانیه به جلو می‌رود. سپس هر پنجره باعث تولید یک ارایه ۴۰ عددی از طیف صوت می‌شود و در مجموع برای یک ثانیه صوت ۴۹ سطر ایجاد شده و در یک آرایه دو بعدی با ابعاد ۴۰ در ۴۹ ذخیره می‌شود.

۳.۱.۵- شروع آموزش

بعد از ایجاد محیط برای شروع آموزش شبکه، لازم است که به مسیر پروژه رفته و سپس با استفاده از دستور زیر میتوان آموزش را شروع کرد.

Python train.py –data_dir=./downloaded_data –train_dir=./train_directory –
wanted_word=[‘yes’, ‘no’]

دستور بالا باعث شروع آموزش شبکه می‌شود. این اسکریپت به صورت خودکار داده‌ها را از دیتابیس گوگل دانلود کرده و آموزش سیستم را با با مقادیر پیش‌فرض شروع می‌کند. حجم مجموعه داده زیاد می‌باشد و بسته به سرعت اینترنت ممکن است که زمانی طولانی لازم باشد. این مجموعه داده از این پس بر روی سیستمی که با آن اموزش داده می‌شود خواهد ماند و برای کلمه‌های دیگر میتوانید در تنظیمات ورودی اسکریپت آرایه مختلفی از کلمات را به آن بدهید.

تنظیمات دیگری هم در هنگام شروع آموزش میتوان به برنامه داد که تعدادی از آن را بررسی میکنیم:

جدول (۱-۳) - تنظیمات کلی برای آموزش مدل

Option Command	Option functionality	Example
window_stride	قدم‌های جلو رونده پنجره	۲۰
silence_percentage	درصد تعداد صداهای ساکت ورودی	۲۵
unknown_percentage	درصد کلمات نا مشخص	۲۵
how_many_training_steps	تعداد قدم‌های آموزش شبکه	۲۰
learning_rate	ضریب آموزش شبکه	۰.۰۰۰۱

۳.۱.۶- خروجی شبکه

خروجی شبکه بعد از لایه‌ی SoftMax می‌باشد. شبکه‌ی موجود ۴ خروجی به شکل‌های Silence, Unknown, Yes و No دارد. برای هر کدام از این کلمات که به صورت یک صوت یک ثانیه‌ای به شبکه داده شده‌اند یک میزان امتیازی در نظر گرفته می‌شود که بین ۰ تا ۲۵۵ است. برای مثال خروجی شبکه می‌تواند به شکل زیر باشد:

[10, 10, 231, 80]

که با توجه به داده‌های ورودی پیش‌بینی می‌کند که کلمه‌ی ورودی گفته شده Yes می‌باشد. با توجه به اینکه ورودی شبکه ۱ ثانیه گذشته می‌باشد، باید سیستم هر ۱ ثانیه ورودی‌ها را چک کند و نتایج را اعلام کند. اما این کار درست نیست، چون شناساگر خوب شناساگری هست که بتواند به با دریافت قسمتی از کلمه آن را تشخیص دهد. برای همین در هر ثانیه در سیستم سخت افزاری ۱۰ بار در ثانیه این شناسایی انجام می‌شود و در صورت شنیدن کلمه‌ای آن را نمایش می‌دهد.

۳.۱.۷- خروجی گرفتن از مدل آموزش دیده

هدف آموزش شبکه، پیدا کردن وزن‌های بین لایه‌هاست. تنسورفلو نیز بعد از اتمام آموزش وزن‌های بدست آمده را بر اساس روش‌های بازگشتی در فولدری در مسیر train_dir که در شروع آموزش به آن معرفی کرده‌ایم قرار میدهد. این فایل در زیر فولدر speech_commands_train ذخیره شده که تمام اطلاعات شبکه مورد نظر را در خود جای داده است. این برنامه به ازای هر ۱۰۰۰ قدمی که جلو میرود یک نقطه ثبت قرار میدهد و وزن‌های شبکه را ذخیره می‌کند تا اگر به صورت ناخواسته خطایی در برنامه صورت گرفت آموزش را از نقطه ثبت شده ادامه دهد و اطلاعات از دست نرود.

حال میخواهیم مدلی که به صورت خام تنسورفلو ذخیره کرده است را به صورت فشرده و با فرمت استاندارد شبکه‌های عصبی که به صورت pb می‌باشد ذخیره کنیم. برای این کار ابتدا ترمینال سیستم‌عامل خود را به محل گفته شده برده و دستور زیر را اجرا میکنیم:

```
python freeze.py –output_file=../../OutPutModel.pb
```

این دستور به صورت خودکار نقاط ثبت شده توسط برنامه را شناسایی کرده و آخرین نقطه را دریافت و خروجی مدل را در فایلی به نام OutPutModel.pb ذخیره می‌کند. این فایل کامل مدل آموزش داده شده است که میتون با استفاده از تنسورفلو به راحتی آن را اجرا کرد و به فرمت‌های دیگر تبدیل کرد.

۳.۱.۸- تبدیل به فرمت TFLite

برای تبدیل به فرمت TFLite که مدل بسیار کم حجم برای استفاده در سیستم‌های کم توان و بهینه است، با یک دستور میتوان این عمل را انجام داد. این دستور به صورت پیش‌فرض همراه تنسورفلو بر روی سیستم نصب می‌شود. ابتدا ترمینال را در مسیری که OutPutModel.pb برده و با دستور زیر میتوان تبدیل را به راحتی انجام داد:

```
toco –graph_def_file=OutPutModel.pb –output_fule=OutPutModel.tflite –
input_shapes=1,49,40,1
```

با استفاده از دستورات ترمینال یا به صورت گرافیکی میتوان حجم فایل نهایی را بدست آورد که برای این پروژه در حدود ۱۸ کیلوبایت می‌باشد که بسیار فایل سبکی است.

۳.۱.۹- تبدیل به آرایه C

برای استفاده از TFLite در سیستم‌های نهفته نیاز است که خروجی شبکه به صورت یک فایل تحت زبان برنامه‌نویسی C تبدیل شود. فایل TFLite به صورت باینری ذخیره شده است که به راحتی در سیستم عامل لینوکس میتوان با دستور پیش‌فرضی که وجود دارد آرایه C سیستم را به راحتی دریافت کرد. دستور زیر این تبدیل را انجام میدهد.

```
xxd -i OutPutModel.tflite > ./out_put_model.cc
```

این دستور آرایه‌ای از شبکه ایجاد می‌کند که شکل کلی آن در عکس زیر نمایش داده شده است:

```
unsigned char _content_tiny_conv_tflite[] = {
  0x1c, 0x00, 0x00, 0x00, 0x54, 0x46, 0x4c, 0x33, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x0e, 0x00, 0x18, 0x00, 0x04, 0x00, 0x08, 0x00, 0x0c, 0x00,
  // ...
  0x00, 0x09, 0x06, 0x00, 0x08, 0x00, 0x07, 0x00, 0x06, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x04
};
unsigned int _content_tiny_conv_tflite_len = 18208;
```

شکل (۴-۳) - خروجی نهایی فایل cc که میتوان در میکروکنترلر استفاده کرد.

۳.۲- معماری سخت‌افزار سیستم

۳.۲.۱- آماده سازی محیط برنامه‌نویسی

با توجه به توضیحات فصل قبل در این پروژه از سیستم‌عامل Mbed بر روی میکروکنترلر استفاده شده است. به همین منظور برای برنامه نویسی و کامپایل کردن برنامه نوشته شده نیاز است که یک محیط برنامه نویسی مناسب انتخاب شود.

برای سیستم‌عامل‌های ویندوز و مک، شرکت Mbed به صورت اختصاصی برنامه‌ای تحت عنوان Mbed Studio ایجاد کرده است. با جست‌وجوی ساده میتوان برنامه رایگان را دانلود و استفاده کرد. اما برای سیستم‌عامل لینوکس، یک افزونه برای محیط برنامه نویسی VSCode به نام PlatformIO موجود است. در این پروژه از این محیط برنامه نویسی استفاده شده است. این محیط را میتوان با نصب VSCode در ابتدا و سپس در قسمت افزونه‌ها با جست‌وجو نام این افزونه آن را پیدا و سپس نصب کرد.

همچنین برای ارتباط با برد و خطا آزمایی آن گروه Mbed یک محیط دستور خطی[1] ایجاد کرده اند که میتوان به صورت برخط داده ها و متغیرهای موجود بر روی حافظه میکروکنترلر را خواند.

نمونه‌ای از محیط برنامه نویسی تحت PlatforIO را میتوانید در شکل زیر مشاهده کنید:

۳.۲.۲- بخش‌های سیستم سخت‌افزاری

برای پیاده سازی یک شبکه‌ی عصبی بر روی میکروکنترلر نیاز است که معماری نرم‌افزاری - سخت‌افزاری داشته باشیم و سیستم را به صورت ماژولار پیکربندی کنیم. به همین منظور ساختار زیر در پروژه استفاده شده است:

[1]Command Line Interface

۱.Audio Provider : جمع آوری داده های صوتی محیطی با استفاده از سنسور مورد نظر (داده‌های ورودی)

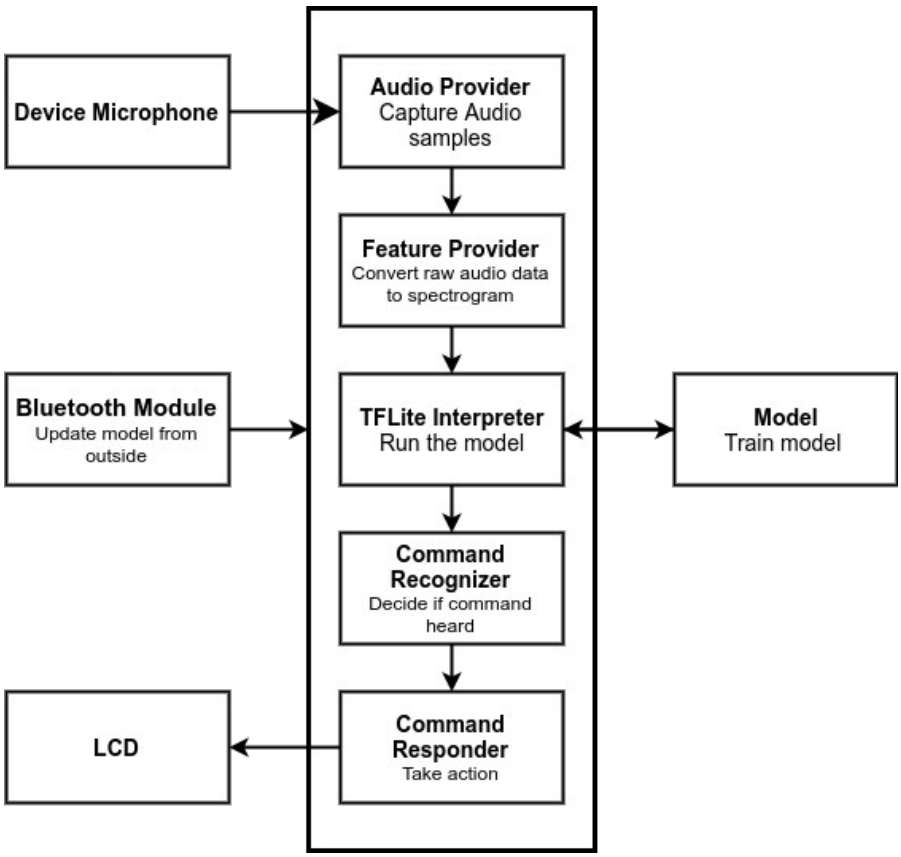
۲.Feature Provider : پیش پردازش داده‌های ورودی و استخراج ویژگی‌های مورد نیاز شبکه عصبی

۳.TF Lite Interpreter : اجرای شبکه بر روی ویژگی های پردازش شده

۴.Command Recognizer : پس‌پردازش و بدست آوردن خروجی و مشخص کردن دقت‌ها

۵.Command Responder : استفاده از نتایج و برای تصمیم گیری

دیاگرام زیر مراحل بالا را به صورت شکلی نمایش می‌دهد:



شکل (۵-۳) - دیاگرام اجزای سیستم سخت افزاری

حال به بررسی اجزای سیستم به صورت تفکیک شده میپردازیم:

۳.۲.۳- ماژول اصلی سیستم Main

در این سیستم یک ماژول اصلی به نام Main مسئول گردآوری تمامی قسمت های مختلف و اتصالات آنها به یکدیگر می‌باشد. معمولا در میکروکنترلرها این ماژول به صورت یک فایل در ساختار برنامه قرار دارد و تابع اصلی برنامه یعنی Main را شامل می‌شود. در این تابع یک حلقه بینهایت وجود دارد که برنامه همیشه در حال اجرا بر روی سیستم می‌باشد.

این ماژول در فایل main.cc در محل اصلی پروژه قرار دارد. در این ماژول ابتدا تمامی ماژول‌های دیگر ایجاد شده و ارتباط بین آنها مشخص می‌شود.

۳.۲.۴- ماژول Audio Provider

این ماژول مسئول جمع آوری داده‌ها از میکروفون و ذخیره آنها بر روی حافظه اصلی میکروکنترلر می‌باشد. هسته اصلی این ماژول تابع GetAudioSamples می‌باشد. این تابع با گرفتن ورودی هایی همچون زمان شروع، مدت زمان و اندازه نمونه ها داده‌های میکروفون را خوانده به صورت یک آرایه بازگردانی می‌کند. ارایه داده ها به صورت ۱۶ بیتی Pulse Code Modulation که یک روش مرسوم و ساده نمونه برداری از سیگنال آنالوگ است. نکته‌ای که وجود دارد این است که تمام ورودی ها به صورت پوینتر به تابع داده می‌شوند و تابع بعد از نمونه برداری، مقادیر آن‌ها را به روز کرده تا برای نمونه برداری های بعدی استفاده شوند.

این ماژول با سرعت ۱۶ هزار در ثانیه نمونه برداری می‌کند. با توجه به اینکه سیگنال صوتی انسان به صورت میانگین در ۳ کیلوهرتز می‌باشد و نرخ نایکوئیست مورد نیاز با این فرکانس ۶ کیلوهرتز می‌شود و همچنین سرعت پردازنده، این مقدار انتخاب شده است که خیلی بیشتر از نرخ نایکوئیست مورد نیاز است. این ماژول به صورت polling کار میکند و در هر بار فراخوانی تعداد داده‌های خواسته شده را خوانده در آرایه مورد نظر بازگردانی می‌کند.

در این پروژه از کتابخانه‌های بسیاری تحت فریموورک Mbed استفاده شده است. یکی از این کتابخانه‌ها برای اتصال و پیکر بندی میکروفون موجود بر روی برد آموزشی است. این کتابخانه به نام Audio_DSIC_OF746NG می‌باشد که در سایت خود Mbed قابل دریافت است.

برای راه اندازی میکروفن تنظیماتی لازم است که در فایلی به نام audio_provider.cc قرار دارد که میتوان آن‌ها را تغییر داد. برای مثال در این پروژه میزان فرکانس نمونه برداری با توجه به مقادیر لازم ۱۶ هزار نمونه و اندازه بافر لازم که اطلاعات بر روی آن ذخیره می‌شود به اندازه ۲۰۴۸ خانه ۳۲ بیتی در نظر گرفته شده اند.

۳.۲.۵- ماژول Feature Provider

ماژول Feature Provider داده‌های خام دریافتی از میکروفون را دریافت و به طیف‌سنج‌های ۱ ثانیه‌ای که خوراک مدل هستند، تبدیل می‌کند. این ماژول به صورت یک کلاس در کد تعریف شده است. در هنگام ساخت کلاس نیاز است که تعداد داده‌هایی که برای ایجاد طیف‌سنج نیاز است و پوینتر حل داده ها به آن داده شود.

در ابتدا لازم است ساختار توضیح داده شود. داخل برنامه هر طیف‌سنج یک آرایه دوبعدی می‌باشد که ۴۰ ستون و ۴۹ سطر دارد. هر سطر مسئول ۳۰ میلی‌ثانیه از داده‌ها که در ۴۳ فرکانس تقسیم بندی شده‌اند میباشد. این اعداد بر طبق مقاله‌ی بایدو استفاده شده اند.

برای ایجاد هر سطر، سیستم ۳۰ میلی ثانیه برشی از صدا را به ورودی یک FFT میدهد این الگوریتم ورودی را به توزیع فرکانسی ۲۵۶ تایی بین ۰ تا ۲۵۵ (۸ بیت) تبدیل می کند. سپس برای تبدیل به طیف سنج، هر ۶ تای آنها را میانگین گرفته و به ۴۳ گروه تبدیل میکند.

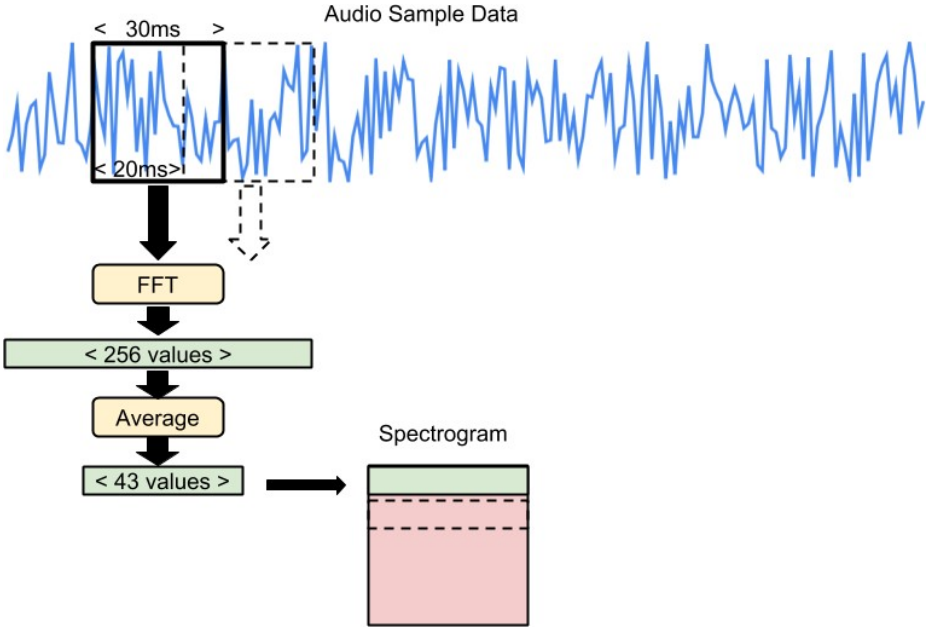
حال برای ساخت یک طیف‌سنج ۱ ثانیه‌ای، هر ۳۰ میلی ثانیه برش را روی یک سطر قرار میدهیم. از سطر ۲ به بعد هر برش از صدا ۱۰ میلی‌ثانیه با برش قبلی همپوشانی دارد. به همین منظور ۴۳ سطر ما داده از صدا خواهیم داشت.

با توجه به اعداد گفته شده نیاز است که تعداد نمونه‌هایی که Audio Provider نیاز دارد را بدست

آوریم. این تعداد با توجه به پنجره های ۳۰ میلی ثانیه‌ای برابر است با:

$$\text{تعداد نمونه‌های ورودی} = \frac{\text{مدت زمان ضبط} \times \text{نرخ نمونه‌برداری}}{\text{ثانیه}} = \frac{30ms \times 16000Hz}{1000ms} = 480$$

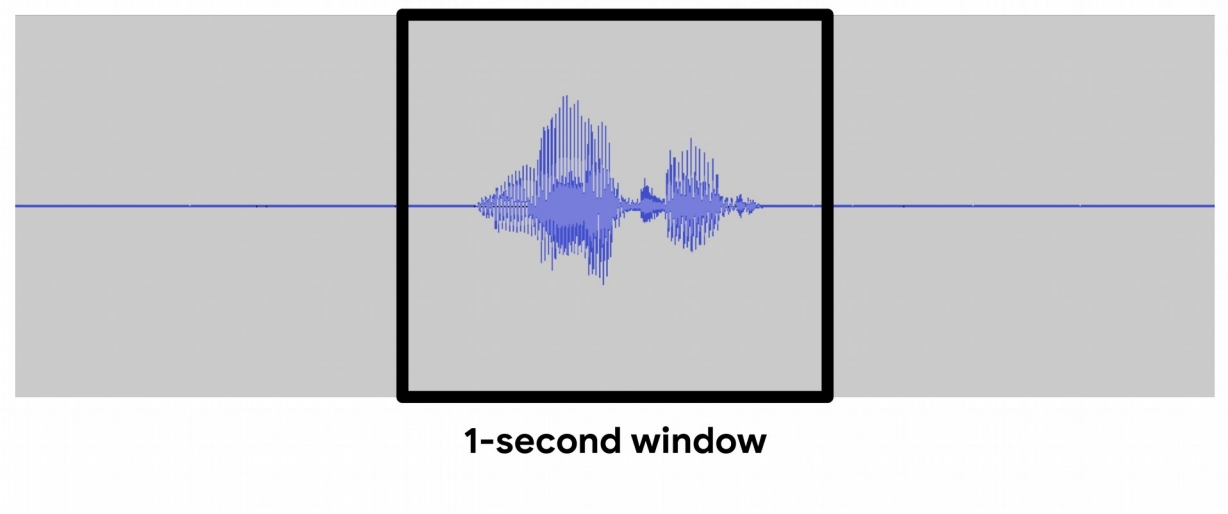
این تعداد نمونه‌هایی است که در هر بار اجرای Audio Provider نیاز است که یک پنجره ۳۰ میلی ثانیه‌ای از صدا را دریافت کند. اما همانطور که توضیح داده شد در فصل قبل بدلیل اینکه الگوریتم FFT نیاز دارد که توانی از ۲ نمونه دریافت کند، تعداد نمونه‌ها را بیشتر میکنیم و تعداد ۵۱۲ نمونه از ورودی دریافت میکنیم و سپس فرکانس‌هایی که گوش به آنها حساس نیست را با استفاده از الگوریتم MFCC خارج میکنیم.



شکل (۶-۳) - روند تبدیل صدا به طیف‌سنج

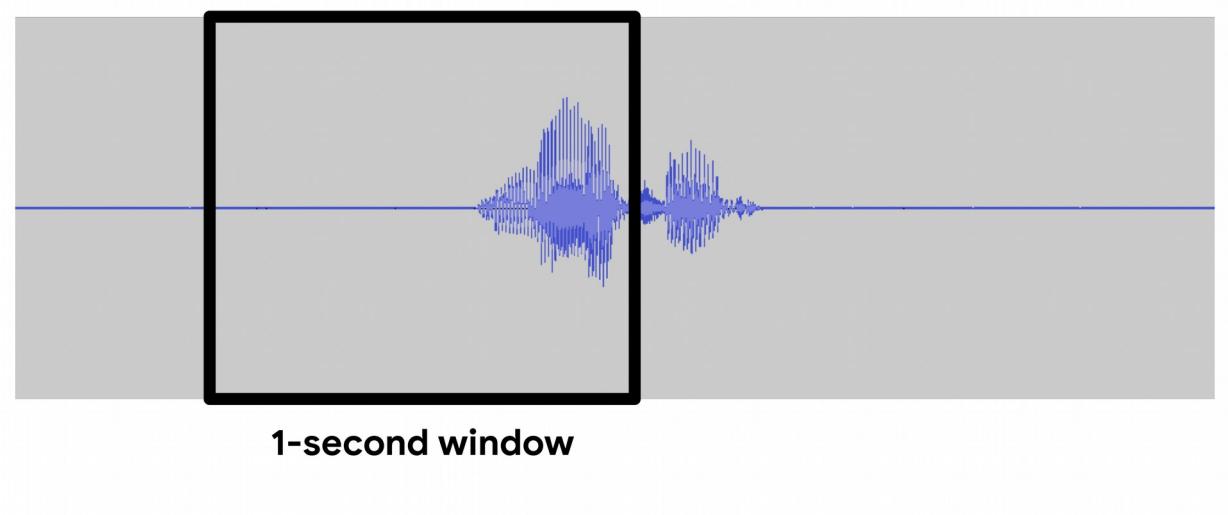
۳.۲.۶- ماژول Command Recognizer

خروجی مدل به صورت مجموعه‌ای از احتمالاتی است که از آخرین ۱ ثانیه گفتار بدست آمده است. وضیفه این ماژول این است که متوجه شود که چه کلمه‌ای شنیده شده است. راه راحت این است که هر کلمه‌ای که احتمال آن بیشتر بود کلمه‌ای است که شنیده شده است، اما در دنیای واقعی شناسایی گفتار به همین راحتی نیست. برای مثال شکل زیر طیف کلمه noted مشخص شده است:



شکل (۷-۳) - کلمه noted در یک قانیه پرداشت شده است.

حال مدل ما کلمه no را یادگرفته است که در بخش اول کلمه‌ی noted حضور دارد. اگر مدل قسمت اول صوت را نگاه کند کلمه را no تشخصی می‌دهد که درست نیست.



شکل (۸-۳) - قسمت اول کلمه noted که سیستم به اشتباه no تلقی میکند.

برای رفع این موضوع، نیاز است که یک ماژول عملیات تشخیص کلمه را انجام دهد. در اینجاست که Recognizer نقش خود را در سیستم نشان میدهد. در متن برنامه برای برد، در فایل recognize_commands.h وجود دارد که مسئولیت مشخص کردن کلمه را بر اساس ورودی‌های قبلی دارد. در یک آرایه به تعداد ۱۰۰۰ عدد، خروجی‌های شبکه در آن قرار می‌گیرد. سپس با استفاده از یک حلقه، بررسی می‌شود که بیشترین تعداد احتمال صوت برای چه کلمه‌ای بوده است. سپس خروجی را بر اساس ۰ تا ۲۵۵ است در یک ذخیره می‌کند و به تابع بعدی پاس می‌دهد.

زمانیکه کلمه جدیدی را تشخیص داد سیستم، یک پرچم[1] بلند می‌کند که به سیستم بگوید کلمه‌ی جدید را تشخیص داده است. این ماژول شاید یکی از مهم‌ترین ماژول‌ها بوده و دقت سیستم را میتوان با این

[1]Flag

ماژول بهتر کرد.

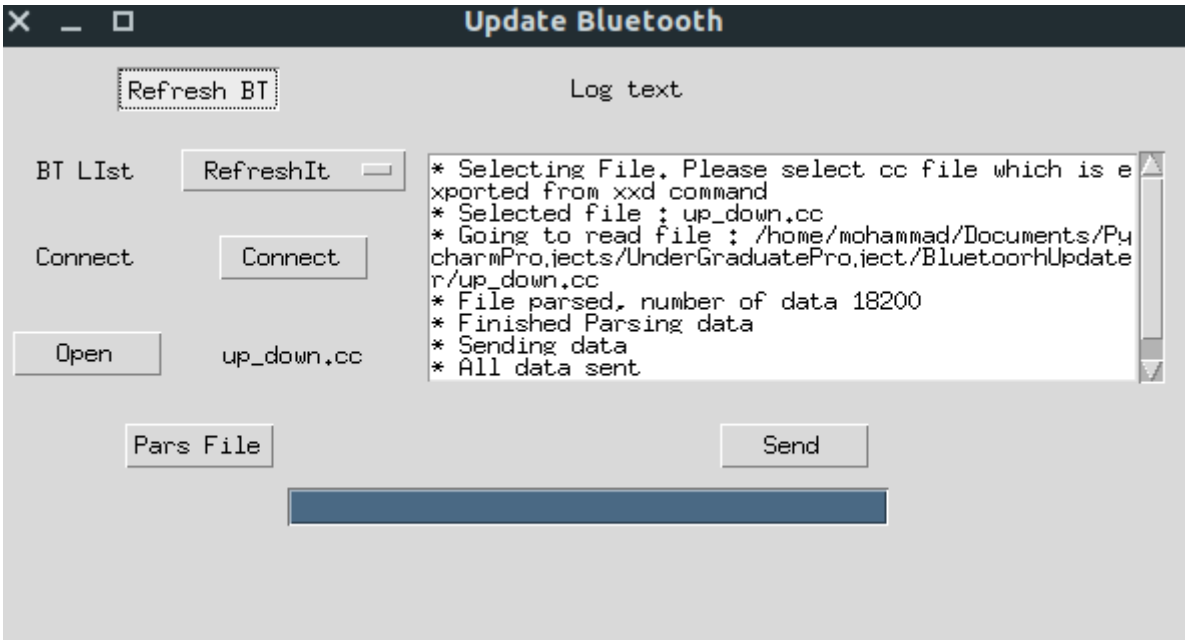
۳.۲.۷- ماژول Command Responder

بعد از تشخصی کلمه لازم است که عملیاتی در سیستم انجام شود. این عملیات بـه صـورت کلی هـدف ایجاد این سیستم بوده است. در این پروژه بعد از تشخیص کلمه، آن را بر روی یک نمایشگر نمایش میدهد. اما در سیستم‌های صنعتی میتوان کارهای دیگری مانند روشن کردن یک سیستم اطلاع رسانی بـا تـوان مصـرفی بالا، ارسال اطلاعات به صورت بدون سیم به سرور ابری و ... انجـام داد. این مـاژول بسـته بـه نیـاز و تعریـف سیستم میتواند متفاوت باشد. در این سیستم به نمایش کلمه بر روی یک نمایشگر بسنده کرده‌ایم.

۳.۲.۸- ماژول bluetooth

در ابتدا سیستم با استفده از مدل آموزش داده شده‌ای که به صـورت پیش‌فـرض بـر روی آن قـرار دارد شروع به کار میکند. اما ممکن است که به صورت بر خط نیاز باشد که مدل را به روز رسانی کـنیم . بـه همین دلیل در کنار سیستم یک ماژول بلوتوث نیز قرار گرفته است که مسئولیت به روز رسانی سیسـتم را بـر عهـده دارد. این ماژول HC-05 که از طریق درگاه UART به برد متصل شده است، از طریق یک رابط گرافیکی که با زبان پایتون نوشته است کنترل می‌شود.

در فصل قبل به تفضیل در مورد آموزش مدل و خروجی گرفتن از آن توضـیح داده شـد. در انتهـا فایـل تولید شده به زبان c که با فرمت .cc خروجی گرفته می‌شـود را میتـوان بـه رابـط کـاربری بـر روی کـامپیوتر شخصی داد و بعد از اتصال رایانه به بلوتوث دستگاه میتوان مدل جدیـد را ارسـال کـرد و از آن زمـان بـه بعـد سیستم با مدل جدید کار می‌کند.



شکل (۹-۳) - رابط گرافیکی برای ارسال اطلاعات

با استفاده از دکمه Refresh BT میتوان بلوتوث مورد نظر را پیدا کرد و سپس به آن متصل شد. سپس فایل خروجی گرفته شده را باز کرده و با زدن دکمه Pars File اطلاعات آن پردازش می‌شود. سپس میتوان با دکمه Send اطلاعات را به برد انتقال داد و برد به صورت اتوماتیک به روز می‌شود.

۳.۳- خلاصه و جمع‌بندی

در این فصل با جزئیات به بررسی سیستم و نحوه پیاده‌سازی آن پرداختیم. سیستم از ۲ بخش نرم‌ازاری و سخت افزاری تشکیل شده است که در بخض نرم‌ازاری هدف ایجاد مدل با دقت بالا برای پیاده‌سازی در قسمت سخت‌افزاری سیستم می‌باشد. در قسمت سخت‌افزاری نیاز است که ماژول‌های متعدد در کنار یکدیگر کار کنند تا بتوانند از صدای ورودی خالص به خروجی و انجام عملیات مورد نظر برسند.

سیستم یکپارچه نرم‌افزاری و سخت‌افزار یکی از سخت‌ترین سیستم‌های موجود میباشد و بدلیل برنامه نویسی سطح پایین سخت‌افزار، این پیچیدگی چند برابر نیز می‌شود.

فصل ۴

اجرا و خروجی‌ها

در فصل قبل پیکربندی سیستم و توضیحات قسمت‌های مختلف آن داده شد. با دانش فصل قبل باید بتوان سیستم را با کدهای موجود دوباره پیاده‌سازی کرد و تغییرات لازم را بر روی آن داد. در این فصل به اجرا و مشاهده نتایج حاصله میپردازیم و توضیح میدهیم که چگونه سیستم خروجی می‌دهد. در ابتدای فصل به خروجی حاصل از مدل آموزش داده شده بر روی سیستم قوی را توضیح میدهیم و در ادامه آن خروجی گرفته شده از سخت‌افزار را شرح می‌دهیم.

۴.۱- نتایج و دقت مدل

در فصل قبل نحوه‌ی آموزش مدل بررسی شد. در این قسمت نتایج بدست آمده را بررسی میکنیم.

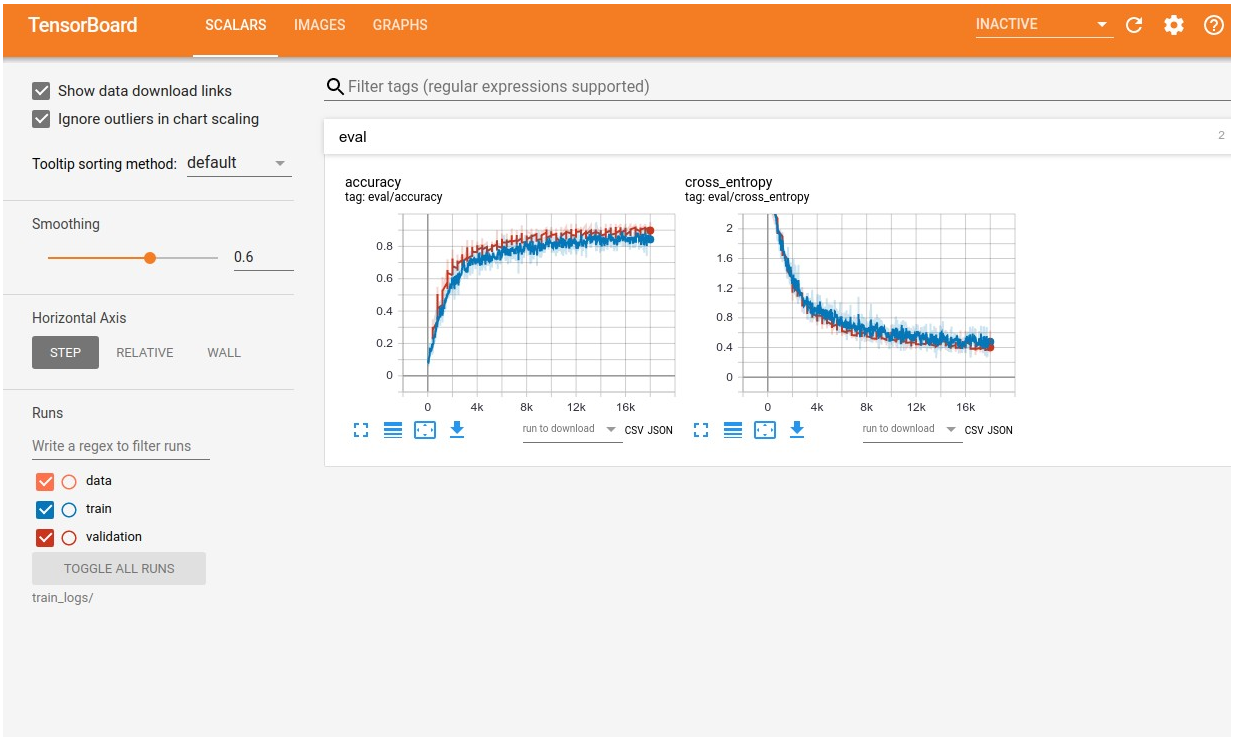
برنامه نوشته شده از یکی از بهترین ویژگی‌های تنسورفلو TensorBoard پشتیبانی می‌کند. سیستم دنبال کننده تنسورفلو و استفاده از نتایج ثبت شده این امکان را میدهد که بتوان به صورت گرافیکی نتایج را مشاهده کرد. با استفاده از دستور زیر میتوان بعد از پایان آموزش مدل نتایج دقت را بررسی کرد:

tensorboard –logdir=train_logs

این دستور محیط گرافیکی را باز می‌کند که میتوان به راحتی نتایج را در آن مشاهده کرد.

برای انتخاب مدل چندین شبکه‌ی مختلف با توجه به محدودیت زمان آزمایش شد. متاسفانه سخت‌افزار مناسب برای آموزش شبکه‌ها موجود نبود و لازم بود برای آموزش هر شبکه در حدود ۱۰ ساعت زمان سپری شود.

نتایج شبکه اصلی و نهایی ما که در قسمت قبل توضیح داده شد به شکل زیر است:

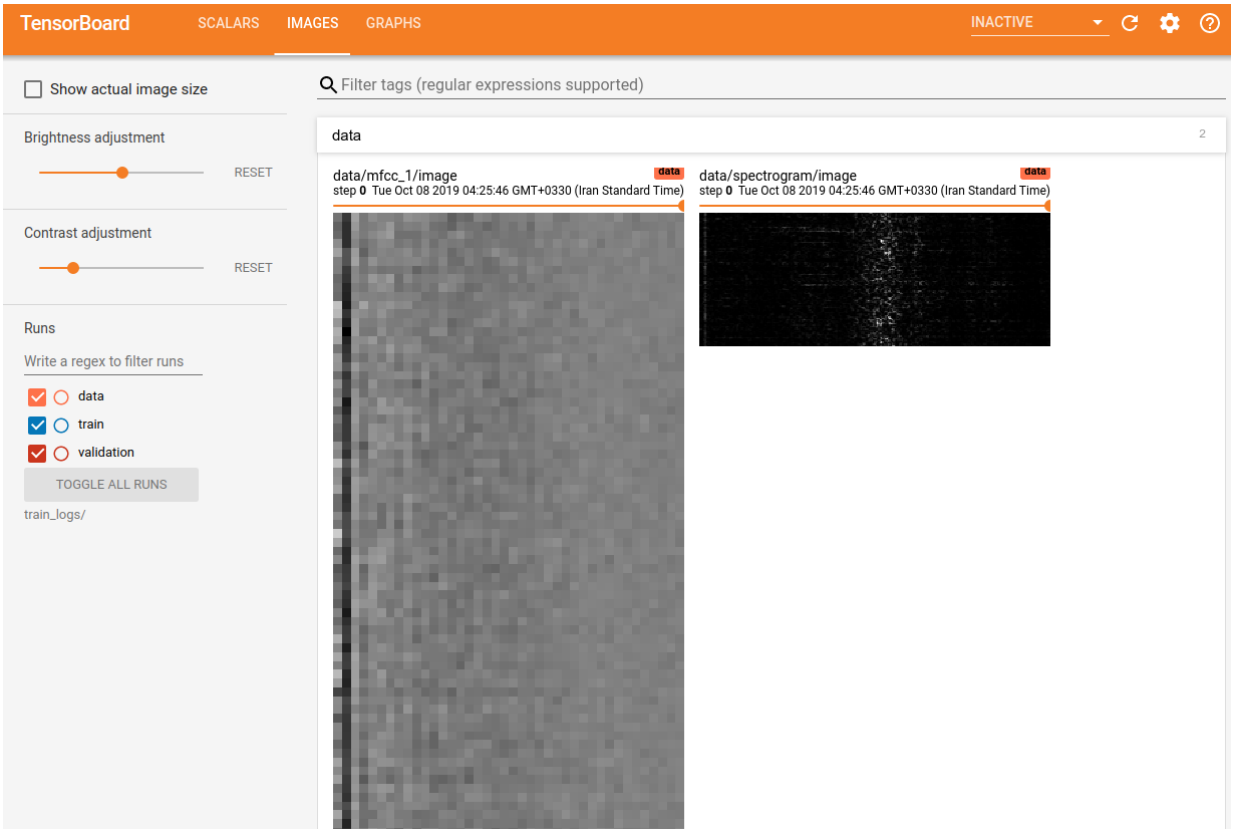


شکل (۱-۴) - نتایج مدل اصلی

دقت نهایی شبکه بالا برای کلمات به ۸۸ درصد رسید. این میزان دقتز بهترین ساختار شبکه‌ای بود که با قیود سازگار بود و امکان قرار گیری آن بر روی میکرکنترلر وجود داشت.

مدل‌های مختلف دیگری نیز طراحی و آزمایش شد. به عنوان مثال یک لایه تمام اتصال دیگر به شبکه اضافه شد، دقت نهایی آن ۸۹ درصد بود اما حجم خروجی آن ۱۰۰ کیلوبایت بود که در مقایسه با حجم خروجی ۱۸ کیلوبایتی مدل اصلی خیلی تفاوت داشتند. همچنین میزان فلش لازم بر روی میکروکنترلر وجود نداشت تا داده‌ها بر روی آن قرار گیرد.

همچنین ورودی تصویر شبکه را نیز این سیستم به ما نمایش میدهد.



شکل (۲-۴) - تصاویر ورودی به شبکه

۴.۲- خروجی سخت‌افزار

بعد از کامپایل کردن کد و ارسال آن به میکروکنترلر، میتوان به صورت UART نتایج را بـر روی پـورت سریال بود مشاهده کرد. همچنین بر روی LCD برد نیز نتایج مشخص می‌شوند.

```
Heard: yes (70)
Heard: silence (64)
Heard: unknown (66)
Heard: no (143)
Heard: silence (64)
Heard: no (83)
Heard: yes (92)
Heard: unknown (66)
Heard: no (78)
Heard: silence (64)
Heard: unknown (66)
Heard: yes (150)
Heard: silence (64)
```

شکل (۳-۴) - خروجی سریال پروژه و مقادیری که شناسایی کرده است.

شکل (۴-۴) - خروجی سیستم به صدای کم توان.

شکل (۴-۵) - خروجی سیستم در زمان شنیدن کلمه no

شکل (۴-۶) - خروجی سیستم در زمان شنیدن کلمه yes

۴.۳- خلاصه و جمع‌بندی

پروژه به خروجی نهایی خود رسید و توانستیم تمامی اجزا را در کنـار همـدیگر قـرار داده و خـروجی را دریافت کنیم.

فصل ۵

جمع‌بندی، نتیجه‌گیری و پیشنهادها

۵.۱- جمع‌بندی

در این تحقیق در گام نخست ابتدا سعی کردیم که نیاز صنعت و روند تکنولوژی را دریابیم و سپس بر اساس نیازی که در کشور حس می‌شود، پروژه‌ای را طرح کنیم که بتوان در پیشرفت روز افزون کشور عزیزمان مؤثر باشد. شرکت‌های ایرانی با تلاش روز افزون خود در تلاش هستند که بتوانند با نوآوری و انجام پروژه‌های سودمند با همراهی دانشجویان توانمند در جهت سرفرازی کشور حرکت کنند.

این پروژه با توجه به نیاز صنعت به سیستمی برای فعال سازی اولیه یک سامانه نهفته و تلاش برای افزایش حریم خصوصی و کاهش مصرف انرژی و با بهره گیری از دانش روز دنیا در زمینه‌های هوش مصنوعی و شبکه‌های عصبی و استفاده از سیستم‌های روز دنیا و میکروکنترلرهای ارزان در عین حال قدرتمند پیاده‌سازی شد.

۵.۲- نتیجه‌گیری

دانش تنهای سخت‌افزاری یا نرم‌افزاری در دنیای امروز دیگر کافی نیست و باید یک مهندس برق و یا کامپیوتر نسبت به دیگر حوزه‌های موجود و مرتبط با توانایی‌هایش آشنایی و در مواقعی توانایی پیاده سازی داشته باشد.

با اتصال دو سیستم جداگانه شبکه‌ی عصبی که برروی یک سیستم بسیار قوی‌تر آمورش داده شده است به یک سخت افزار بسیار ساده و ضعیف، دری به روی پیاده سازی‌های پیشرفته بر روی سیستم‌های توان پایین برای انجام کارهای سطح بالا گشوده شده است.

۵.۲.۱- نوآوری / دستاوردها

با استفاده از این پروژه میتوان روزی در صنعت ایران یک دستیار شخصی کاملاً فارسی داشت که بتوان با نمونه‌های خارجی رقابت کند. در این پروژه ما توانستیم که سیستم‌های مختلف را در کنار هم قرار دهیم و یک سیستم یکپارچه شبکه‌ی عصبی در کنار سخت‌افزار بسیار ساده ایجاد کنیم.

این سیستم انرژی بسیاری پایینی استفاده می‌کند که میتواند در حوزه اینترنت اشیاء کاربر فراوان خواهد داشت. به عنوان مثال میتوان سیستم‌های بسیار کوچکی در خانه تعبیه کرد که بتوانند هر لحظه منتظر شنیدن کلمه‌ی خاصی باشند و در صورت شناسایی با سرور مرکزی ارتباط برقرار کنند و اطلاعات را جابجا کنند. شنود همیشگی این سیستم‌ها دیگر حذف می‌شود و بدلیل پردازش داخلی آن‌ها میتوان از حریم خصوصی افراد نیز محافظت کرد.

۵.۲.۲- محدودیتها

در این پروژه یکی از محدودیت‌های اصلی نبود مستندات کامل و نو بودن حوزه می‌باشد. به همین دلیل در بعضی مواقع نیاز بود که با خود تیم اصلی تنسورفلو در تماس بود. از سوی دیگر، وجود تحریم‌ها بوردهای از ورود بردهای آموزشی به روز و سیستم‌های مخابراتی که میتوان در اینگونه سیستم‌ها استفاده کرد مشکل ساز بود. با این حال با تلاش و همراهی دوستان عزیز توانستیم مقداری از محدودیت‌ها را کم کرده و پروژه را به خروجی برسانیم.

۵.۲.۳- پیشنهادها

شرکت‌هایی که در حوزه اینترنت اشیاء مشغول به کار هستند میتوانند به راحتی از این سیستم استفاده کنند و گجت‌های بسیار کارامدی تولید کنند.

برای دوستانی که میخواهند این پروژه را ادامه یا گسترش دهند، پیشنهاد میکنم که حتماً برای مجموعه داده فارسی این سیستم را آموزش دهند و آزمایش‌های اولیه برای افزایش دقت سیستم را انجام دهند. همچنین کدهای نوشته شده برای برد مورد استفاده بهینه نبوده و بدلیل تنگنای وقت نیاز بود که به یک دقت حداقلی بسنده کنیم. به همین دلیل میتوان با افزایش بازدهی سیستم سرعت آن را افزایش و مصرف توان کلی سیستم را کاهش داد.

همچنین برای تبدیل گفتار به متن فارسی بدلیل سخت بودن زبان فارسی بازار بسیار بزرگی وجود دارد که میتوان به آن ورود کرد و به شرکت‌های مختلف خدماتی را ارائه داد. همچنین سیستم‌هایی نظیر این پروژه به صورت محدود میتوانند کلمات را شناسایی کنند و وجود یکی سیستم ابری که گفتار فارسی را به متن تبدیل کند بسیار کمک کننده خواهد بود.

فصل ۶

مراجع

[1] Hasan, R., Jamil, M., Rabbani, G., and Rahman, S.,“Speaker Identification Using Mel Frequency Cepstral Coefficients”, 3rd International Conference on electrical & Computer Engineering, ICECE 2004, Dhaka, Bangladesh

[2] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, Andrew Y. Ng. *Deep Speech: Scaling up end-to-end speech recognition.*

[3] T. N. Sainath, C. Parada, "Convolutional neural networks for small footprint keyword spotting", Proc. of Interspeech, 2015.

[4] TensorFlow Lite: TensorFlow's lightweight solution for mobile and embedded devices: https://www.tensorflow.org/lite/

[5] Wikipedia. *Artificial neural network.*[online]. Available: URL https://en.wikipedia.org/wiki/Artificial_neural_network

Abstract:

Voice assistance is one of the hottest topics these days. Big companies try to develop their voice assistance which can make it easier for users to use their products. Cell phones now include voice assistance internally. But the phone processors are strong enough to run the ML models. In this project, we try to implement Tensorflow on the microcontroller. STM32F7-Disco is one of the development kits from STM Co. that include a powerful microcontroller with good peripheral to the users in IoT devices. Using Tensorflow lite gives this opportunity to learn Yes, No word with NN. After that convert model to TFLite and use it in the microcontroller.

Keywords:

Voice Assistance, Speech to Text, Embedded Systems, Neural Networks, Microcontrollers

University of Tehran

College of Engineering

School of Electrical and Computer Engineering

Speech Recognition With Microcontrollers

A thesis submitted to the Undergraduate Studies Office

In partial fulfillment of the requirements for
The degree of Electrical Engineering in
Electronic Digital Systems

By:

Seyed Mohammad Hosseini

Supervisor:

Dr. Mahdi Kamal