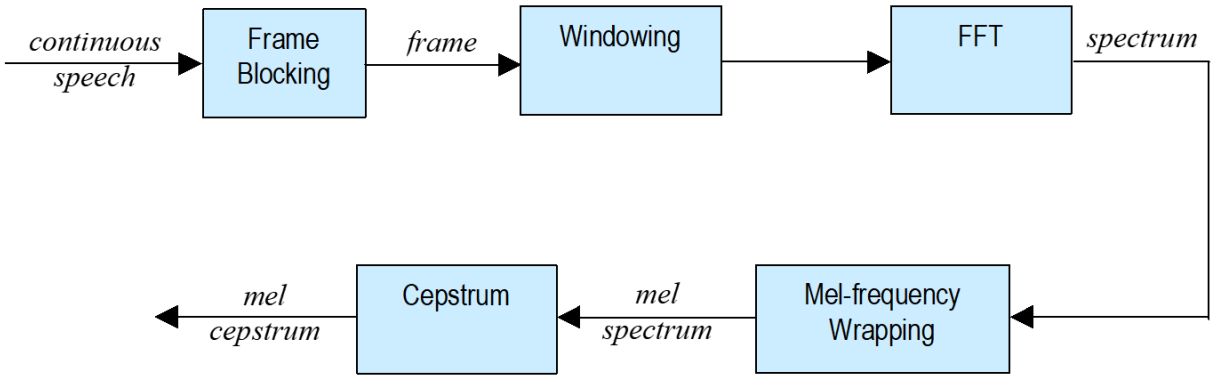
تشخیص گوینده از روی صدا

در این بخش می‌خواهیم با استفاده از ویژگی‌های فرکانسی سیگنال صوت، گوینده را تشخیص دهیم. یک سیستم *speaker recognition* از دو بخش *feature extraction* و *feature matching* تشکیل می‌شود. در بخش *feature extraction* ویژگی‌های مناسب که نماینده‌ی خوبی برای گوینده باشند از سیگنال صدا استخراج می‌شود و در بخش *feature matching* از این ویژگی‌ها برای پیش‌بینی صداهای جدید استفاده می‌شود. در این پروژه ویژگی‌های استخراج شده بردار *MFCC* است و برای طبقه بندی از طبقه‌بندهای *knn* و شبکه عصبی بهره خواهیم برد.

Feature Extraction

Mel-Frequency Cepstrum Coefficients (MFCC): بلوک دیاگرام یک سیستم استخراج *MFCC* در شکل زیر دیده می‌شود. این نوع استخراج ویژگی در واقع عملکرد گوش انسان را تقلید می‌کند و از بخش‌های زیر تشکیل می‌شود:



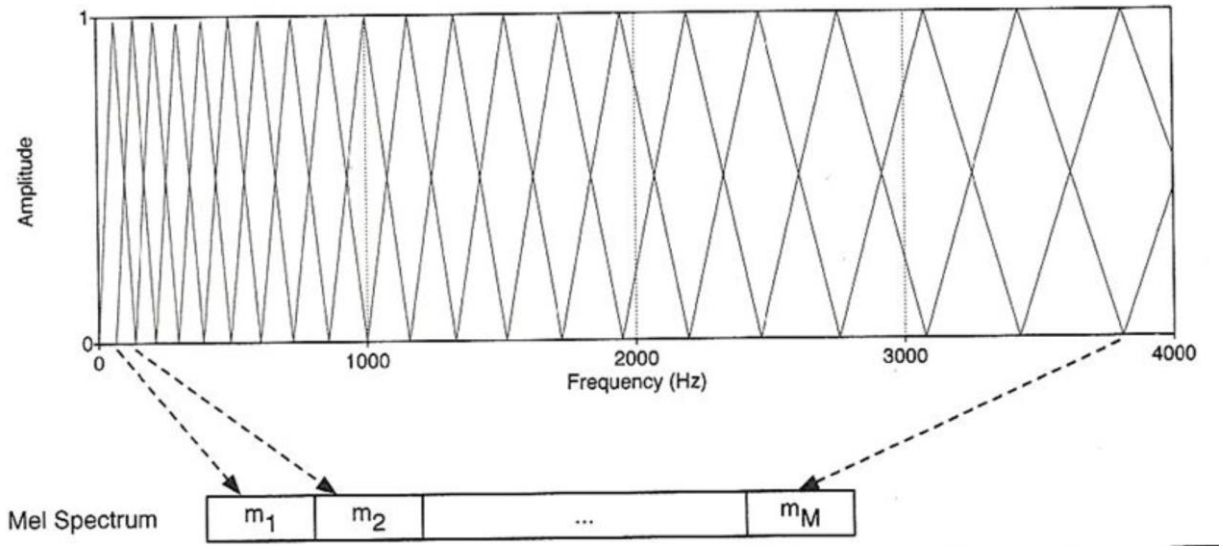
- *Frame Blocking*: در این بخش سیگنال صحبت به فریم‌هایی شامل N نمونه تقسیم می‌شود فریم دوم از نمونه‌ی $M+1$ ام آغاز می‌شود به همین ترتیب بقیه‌ی فریم ها هم به گونه‌ای قرار می‌گیرند که از فریم دوم به بعد، هر فریم با فریم قبل از خود $N-M$ نمونه‌ی مشترک دارد. در این پروژه $N = 256$ و $M = 100$ قرار داده می‌شوند.

- *Windowing*: برای مینیمم کردن ناپیوستگی‌های سیگنال در شروع و پایان هر فریم از پنجره‌گذاری استفاده می‌شود. معمولا برای این کار از پنجره‌ی *hamming* استفاده می‌شود که فرم زیر را دارد. برای اعمال این پنجره از تابع *hamming* متلب کمک بگیرید.

$$w[n] = 0.54 - 0.46\,cos\left(\frac{2\pi n}{N-1}\right) \quad n = 0, \dots, N-1$$

- *FFT*: گام بعدی اعمال *fft* روی هریک از فریم‌ها است. سپس اندازه‌ی *fft* را به توان ۲ رسانده و طیف توان را به دست می‌آوریم. تعداد نقاط *fft* را برای هر فریم برابر با طول سیگنال (همان ۲۵۶) قرار دهید. با توجه به این که خروجی *fft* یک سیگنال متقارن است فقط نیمه‌ی اول آن را نگه دارید.

- *Mel-frequency Wrapping*: درک انسان از صوت به صورت مقیاس خطی صورت نمی‌گیرد، بلکه در فرکانس‌های کمتر نسبت به فرکانس‌های بالا قابلیت تشخیص تغییرات فرکانس بیشتر است. بنابراین در این بخش فرکانس را از مقیاس *Hz* به مقیاس *Mel* تبدیل می‌کنیم. تبدیل به این مقیاس معمولا با استفاده از یک بانک فیلتری انجام می‌شود. بانک فیلتری از تعدادی فیلتر میان‌گذر مثلثی تشکیل می‌شود که هر یک حول یک فرکانس مرکزی قرار گرفته‌اند. این بانک فیلتری در حوزه‌ی فرکانس اعمال می‌شود و مجموع انرژی قرار گرفته داخل هریک از فیلترها محاسبه می‌شود. کافی است هر یک از فیلترها در طیف توان به دست آمده از بخش قبل ضرب شود. به این ترتیب ضرایب *mel spectrum* حاصل می‌شود. فایل *melfb.m* کد متلب مربوط به بانک فیلتری را شامل می‌شود. در پیاده‌سازی از این فایل کمک بگیرید. با توجه به توضیحات موجود در *melfb.m* در پارامترهای تابع، تعداد فیلترها را ۲۰ در نظر بگیرید. سپس برای هر فریم، بانک فیلتری به دست آمده را در طیف به دست آمده از بخش قبل ضرب کنید.



- *Cepstrum*: در آخر *mel spectrum* به دست آمده از بخش قبل لگاریتم گرفته و با استفاده از *Discrete Cosine Transform (DCT)* آن را به حوزه‌ی زمان می‌بریم تا *mel frequency cepstrum coefficients(MFCC)* حاصل شود. به این ترتیب برای هر فریم یک بردار ضرایب *MFCC* به دست می‌آید.

۱) تابع *mfcc.m* را به کمک توابع *wavread, hamming, fft, dct* و *melfb* برای محاسبه‌ی *MFCC* از روی سیگنال صوت ورودی بنویسید. خروجی‌های مراحل *windowing*، *fft* و خروجی نهایی را برای *s1.wav* نمایش دهید.

Feature Matching

- انجام پیش‌بینی در مورد داده‌های *test* که به کدام گوینده تعلق دارند، یک مسئله‌ی طبقه‌بندی است. برای حل این مسئله، بهتر است ابتدا بردارهای ویژگی به دست آمده برای هر یک از گوینده‌ها را به تصویر درآورد.

۲) برای این کار فقط موقعیت ابعاد ۵ و ۶ را از بردارهای *MFCC* به دست آمده برای همه‌ی گوینده‌ها در یک شکل ترسیم کنید. آیا میان نواحی داده دو گوینده‌ی متفاوت همپوشانی وجود دارد؟

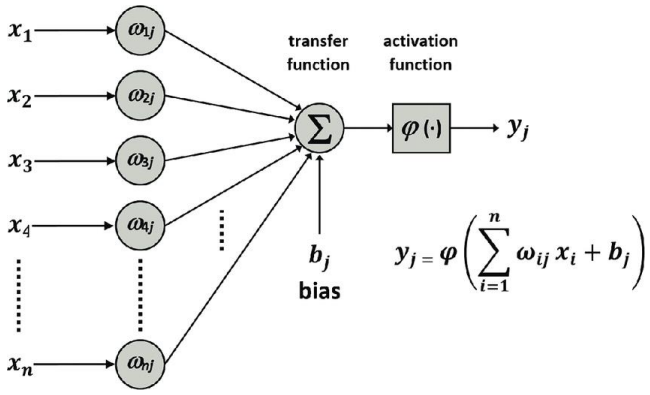
- در بخش‌های بعدی برای این که تعداد فریم‌های تمامی داده‌ها برابر شود، ۱۰۰ فریم وسط هر داده را استفاده می‌کنیم. همچنین از هر بردار *mfcc* فقط ضرایب ۲ تا ۱۳ را از میان ۲۰ ضریب به کار می‌بریم. این کار به این علت انجام می‌شود که بقیه‌ی ضرایب معمولا بهبودی در نتیجه ایجاد نمی‌کنند. همچنین فاصله‌ی بین بردارهای ویژگی دو گوینده به صورت فریم‌های متناظر اندازه‌گیری شود. (فاصله‌ی فریم اول گوینده ۱ با فریم اول گوینده ۲ و ...)

یک روش برای طبقه بندی، استفاده از طبقه بند *k nearest neighbor* یا *knn* است. در این نوع طبقه‌بندی ابتدا فاصله‌ی بردار *test* از تک تک بردارهای *train* محاسبه می‌شود. سپس به تعداد *k* تا از کمترین فاصله‌ها بررسی می‌شوند که مربوط به کدام گوینده از داده‌ی *train* هستند. هر گوینده‌ای که بیشترین تکرار را بین این *k* کمترین فاصله داشته باشد، به عنوان پیش‌بینی انتخاب می‌شود.

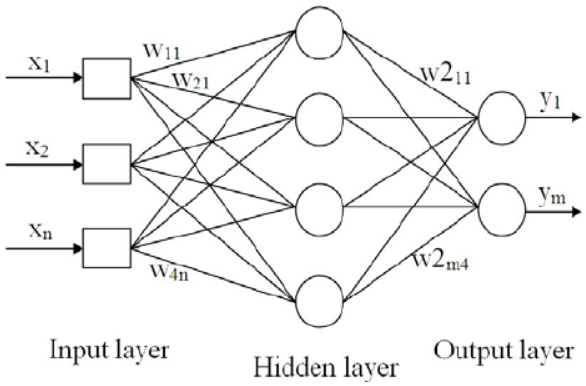
۳) تابع *knn_classifier* را به گونه‌ای بنویسید که *k*، بردارهای *mfcc* مربوط به داده‌ی *train* و داده‌ی *test* را به عنوان ورودی دریافت کند. *k* را به گونه‌ای انتخاب کنید که تمامی پیش‌بینی‌ها درست انجام شود.

شبکه‌ی عصبی مصنوعی *(ANN)* یک مدل محاسباتی است که از سیستم عصبی مغز انسان الهام گرفته شده است. واحد اصلی محاسبات در شبکه‌ی عصبی نورون‌ها هستند. هر نورون تعدادی ورودی و یک خروجی دارد. عملکرد آن

به این صورت است که یک تابع فعال‌سازی بر یک ترکیب خطی از ورودی‌هایش اعمال می‌کند و نتیجه را خروجی می‌دهد.



یکی از اشکال ساده‌ی شبکه عصبی *Multi-Layer Perceptron(MLP)* است که از قرارگیری نورون‌ها در چندین لایه و در هر لایه چندین نورون ایجاد می‌شود. ساختار شبکه‌ی *MLP* در شکل زیر مشاهده می‌شوند. در این پروژه قصد داریم ساده‌ترین شکل شبکه‌ی عصبی یعنی یک شبکه‌ی تک لایه را به کار ببریم.



با توجه به مسئله، شبکه‌ی عصبی به کار رفته ۱۲ ورودی یعنی ضرایب ۲ تا ۱۳ بردار *MFCC* است. خروجی شبکه را نیز ۸تایی در نظر می‌گیریم، به این معنا که هر خروجی که مقدار ماکزیمم را داشت به عنوان پیش‌بینی شبکه در نظر می‌گیریم. آموزش شبکه به این معناست که وزن‌های شبکه (*w*های به کار رفته در ترکیب خطی هر نورون) به نحوی تعیین شوند که شبکه به ازای هر ورودی بردار *mfcc* قادر باشد خروجی صحیح تولید کند.

۴) با استفاده از تابع *feedforwardnet* یک شبکه‌ی عصبی تک‌لایه تعریف کنید. *Hiddensize* بیانگر تعداد نورون‌ها در لایه‌ی میانی است.

مرحله‌ی بعد، آموزش شبکه با استفاده از تابع *train* متلب است. برای این کار لازم است داده‌های *MFCC* مربوط به داده‌ی *train* و همچنین مقادیر صحیح پیش‌بینی *(target)* برای هر بردار *MFCC* به شبکه داده شود.

۵) با استفاده از تابع *train* شبکه را روی داده‌ی یادگیری آموزش دهید. با استفاده از تابع *view* شکل شبکه را نمایش دهید.

راهنمایی: توجه کنید که در ورودی این تابع هر ستون بیانگر یک *sample* از مجموعه داده است. بنابراین ماتریس داده یادگیری ابعاد 800×12 و ماتریس *target* ابعاد 800×8 خواهد داشت.

۶) برای ارزیابی عملکرد شبکه، این بار داده‌های *test* را به شبکه وارد کنید. پیش‌بینی در مورد هر یک از گوینده‌ها را به این صورت انجام دهید که ابتدا برای تک تک فریم‌ها یک پیش‌بینی به دست آورید، سپس پیش‌بینی که بین فریم‌ها تکرار بیشتری دارد را به عنوان پیش‌بینی نهایی شبکه اعلام کنید.

۷) تعداد نورون‌های لایه‌ی میانی را به نحوی تعیین کنید که تمامی پیش‌بینی‌ها درست باشند.

در صورت وجود هر گونه ابهام سوالات خود را از طریق kimiadinashi@gmail.com مطرح کنید.

کدها را در پوشه‌ی جدا ذخیره کرده و در صورت ابهام در کد، کامنت گذاری کنید.

فایل ارسالی را با فرمت زیر ذخیره کنید:

Project_[first_name]_[family_name]_[student_number].zip