

دانشگاه صنعتی خواجه نصیرالدین طوسی
دانشکده مهندسی برق - کروه مهندسی گنتز

درس مبانی سیستم‌های هوشمند پروژه نهایی - تشخیص و طبقه‌بندی آریتمی‌های قلبی از سیگنال ECG

نام و نام خانوادگی	مهندی خلیلزاده
شماره دانشجویی	۹۹۳۲۲۱۳
تاریخ	۱۴۰۳ بهمن ماه



فهرست مطالب

۵		۱ مقدمه
۵	۱.۱ سیگنال ECG
۶	۲.۱ آریتمی‌های قلبی و اهمیت تشخیص آنها
۶	۳.۱ هدف پژوهه
۷		۲ معرفی مجموعه داده آریتمی MIT-BIH
۷	۱.۲ مشخصات و ساختار مجموعه داده
۸	۲.۲ حاشیه‌نویسی و برچسب‌گذاری
۸	۳.۲ انواع آریتمی‌های ثبت شده
۹		۳ پیش‌پردازش
۹	۱.۳ بارگذاری مجموعه داده
۱۰	۲.۳ بهبود کیفیت سیگنال‌ها
۱۳	۳.۳ جداسازی تپش‌ها
۱۵	۴.۳ متعادل‌سازی مجموعه داده
۱۸		۴ مدل‌های پیاده‌سازی شده
۱۸	۱.۴ مدل اول: رگرسیون لاجستیک Logistic Regression
۱۸	۱.۱.۴ علت انتخاب مدل رگرسیون خطی و بررسی نقاط قوت و ضعف آن
۱۹	۲.۱.۴ نحوه پیاده‌سازی و توضیح کد مدل رگرسیون خطی
۲۰	۳.۱.۴ نتایج مدل رگرسیون خطی
۲۲	۲.۴ مدل دوم: Long Short-term Memory (LSTM)
۲۳	۱.۲.۴ علت انتخاب مدل LSTM و بررسی نقاط قوت و ضعف آن
۲۳	۲.۲.۴ نحوه پیاده‌سازی و توضیح کد مدل LSTM:
۲۵	۳.۲.۴ نتایج مدل LSTM
۲۷		۵ نتیجه‌گیری
۲۹		۶ لینک‌های مهم



فهرست تصاویر

۵	سیگنال ECG نرمال و ویژگی‌های آن [۱]	۱
۹	نمونه‌ای از لبیل‌ها	۲
۱۰	سیگنال خام	۳
۱۱	سیگنال بعد از اعمال فیلتر Notch	۴
۱۲	سیگنال بعد از اعمال فیلتر Notch و Average Moving	۵
۱۲	سیگنال بهبود یافته نهایی	۶
۱۵	یک ضربان جدا شده	۷
۱۵	توزیع تعداد اعضای هر کلاس	۸
۱۶	توزیع تعداد اعضای هر کلاس بعد از متعادل‌سازی	۹
۱۶	نمونه ضربان کلاس F	۱۰
۱۷	نمونه ضربان کلاس R	۱۱
۱۷	نمونه ضربان کلاس L	۱۲
۱۷	نمونه ضربان کلاس A	۱۳
۱۸	نمونه ضربان کلاس V	۱۴
۱۸	نمونه ضربان کلاس N	۱۵
۲۰	گزارش طبقه‌بندی - مدل رگرسیون خطی	۱۶
۲۱	ماتریس در هم آمیختگی - مدل رگرسیون خطی	۱۷
۲۱	نمودرا ROC - مدل رگرسیون خطی	۱۸
۲۲	ساختار یک سلول [۷] LSTM	۱۹
۲۵	معماری مدل LSTM استفاده شده	۲۰
۲۶	گزارش طبقه‌بندی - مدل LSTM	۲۱
۲۶	نمودار Accuracy و Loss برای مدل LSTM	۲۲
۲۷	ماتریس در هم آمیختگی - مدل LSTM	۲۳



فهرست جداول

۸	۱	آریتمی‌های اصلی موجود در مجموعه‌داده
۸	۲	سایر انواع آریتمی‌های ثبت شده در مجموعه‌داده

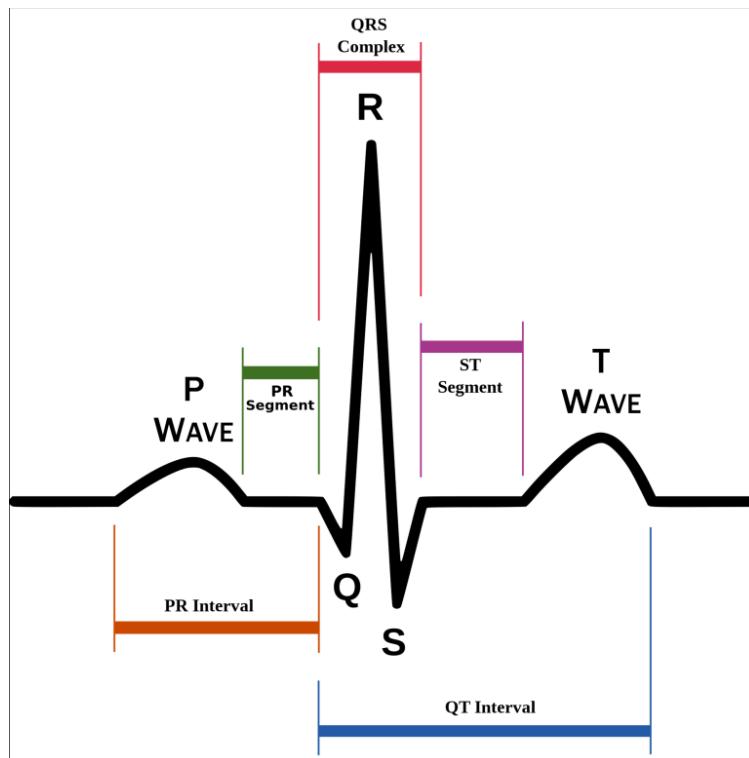


فهرست برنامه‌ها

۹	بخش کد مربوط به بارگیری مجموعه داده	۱
۹	بخش کد مربوط به جداسازی فایل‌ها	۲
۱۰	فیلترها	۳
۱۱	نرمال‌سازی Z-Score	۴
۱۳	جداسازی تپش‌ها	۵
۱۹	بخش کد مربوط به تبدیل داده‌ها به بردار عددی	۶
۱۹	بخش کد مربوط تعریف مدل رگرسیون لاجستیک	۷
۲۳	بخش کد مربوط به آماده سازی داده‌ها برای مدل LSTM	۸
۲۴	بخش کد مربوط پیاده سازی مدل LSTM	۹

۱ مقدمه

۱.۱ سیگنال ECG



شکل ۱: سیگنال ECG نرمال و ویژگی‌های آن [۱]

سیگنال الکتروکاردیوگرام (ECG) ثبت الکتریکی فعالیت قلب است که توسط الکترودهای متصل به سطح پوست اندازه‌گیری می‌شود. این سیگنال اطلاعاتی درباره ریتم قلب، هدایت الکتریکی و سلامت عمومی قلب ارائه می‌دهد. سیگنال ECG از چرخه‌های تکرارشونده‌ای تشکیل شده است که هر چرخه شامل موج P، کمپلکس QRS و موج T است. موج P نشان‌دهنده دپولاریزاسیون دهلیزها، کمپلکس QRS بازتاب دپولاریزاسیون بطن‌ها، و موج T مربوط به ریپلاریزاسیون بطن‌ها است. این مشخصه روی سیگنال در [شکل ۱](#) قابل مشاهده هستند. مطالعه این مؤلفه‌ها برای تشخیص بیماری‌های قلبی از جمله آریتمی، ایسکمی و نارسایی‌های قلبی ضروری است.[\[۲\]](#)

سیگنال ECG دارای ویژگی‌های خاصی است که برای تحلیل و پردازش دقیق آن اهمیت زیادی دارد. یکی از مهم‌ترین ویژگی‌ها نرخ نمونه‌برداری Rate Sampling است که معمولاً بین ۲۵۰ تا ۱۰۰۰ نمونه در ثانیه متغیر است. دامنه ولتاژ نیز بین ۰/۵ تا ۰/۵ میلی‌ولت قرار دارد و تغییرات آن بازتابی از فعالیت الکتریکی قلب است. مدت زمان ثبت سیگنال بسته به نوع ثبت متفاوت است؛ به عنوان مثال، در ECG استاندارد یک ثبت چند ثانیه‌ای انجام می‌شود، در حالی که در ECG هولتر داده‌های ۲۴ ساعته جمع‌آوری می‌شوند. همچنین، سیگنال ECG به دلیل عوامل مختلفی مانند حرکت بیمار، نویز خطوط برق و تداخلات الکتریکی نیاز به پیش‌پردازش و حذف نویز دارد.[\[۲\]](#)



۲.۱ آریتمی‌های قلبی و اهمیت تشخیص آن‌ها

آریتمی‌های قلبی به اختلالاتی در ریتم طبیعی قلب گفته می‌شود که می‌توانند ناشی از مشکلات در سیستم هدایت الکتریکی قلب باشند. این اختلالات شامل ضربان‌های خیلی سریع، خیلی کند یا نامنظم هستند که می‌توانند در اثر عوامل مختلفی از جمله بیماری‌های قلبی، مشکلات ژنتیکی، استرس، مصرف برخی داروها یا عدم تعادل الکتروولیت‌های بدن ایجاد شوند. برخی از انواع آریتمی‌ها مانند فیریالاسیون دهلیزی و انقباضات زودرس بطنی می‌توانند بی‌خطر باشند، اما برخی دیگر مانند تاکی‌کاردی بطنی پایدار و فیریالاسیون بطنی بسیار خطرناک بوده و می‌توانند منجر به ایست قلبی ناگهانی شوند [۴].

تشخیص زودهنگام آریتمی‌های قلبی نقش حیاتی در پیشگیری از عوارض جدی دارد. روش اصلی تشخیص این اختلالات الکتروکاردیوگرام (ECG) است که فعالیت الکتریکی قلب را ثبت کرده و الگوهای غیرطبیعی را شناسایی می‌کند. با این حال، تحلیل دستی داده‌های ECG زمان بر بوده و ممکن است به خطای انسانی منجر شود. به همین دلیل، امروزه از الگوریتم‌های یادگیری ماشین و شبکه‌های عصبی عمیق برای تشخیص خودکار آریتمی‌ها استفاده می‌شود که می‌توانند دقت و سرعت تشخیص را بهبود بخشنند. این مدل‌ها قادرند الگوهای پیچیده ECG را شناسایی کرده و هر ضربان را در دسته‌های مختلفی از آریتمی‌ها قرار دهند [۴].

یکی از مهم‌ترین کاربردهای این فناوری، ادغام مدل‌های تشخیص آریتمی در دستگاه‌های پزشکی هوشمند و سیستم‌های مراقبت از بیمار است. دستگاه‌هایی مانند ساعت‌های هوشمند و مانیتورهای پوشیدنی می‌توانند به طور پیوست ECG بیمار را ثبت کرده و در صورت مشاهده آریتمی، هشدار دهند. این قابلیت به بیماران قلبی و افراد در معرض خطر کمک می‌کند تا در سریع ترین زمان ممکن به پزشک مراجعه کرده و از عوارض جدی جلوگیری کنند. توسعه و بهبود این مدل‌ها می‌تواند تحولی بزرگ در پیشگیری، تشخیص و درمان بیماری‌های قلبی ایجاد کند و میزان مرگ و میر ناشی از آریتمی‌های خطرناک را کاهش دهد [۴].

۳.۱ هدف پژوهه

هدف این پژوهه، تشخیص آریتمی‌های قلبی از روی ضربان‌های ثبت شده در سیگنال‌های ECG با استفاده از پایگاه داده MIT-BIH Arrhythmia است. در این پژوهه، سیگنال‌های ECG بررسی شده و هر ضربان به صورت مجزا تحلیل می‌شود تا مشخص شود که آیا دارای آریتمی است یا خیر. همچنین، هدف این است که بهترین مدل یادگیری ماشین برای تشخیص دقیق این آریتمی‌ها انتخاب شود. با مقایسه عملکرد مدل‌های مختلف، می‌توان به روشنی بهینه برای دسته‌بندی خودکار ضربان‌های قلبی در یکی از پنج کلاس آریتمی موجود دست یافت.



آریتمی‌های قلبی یکی از مهم‌ترین مشکلات پزشکی هستند که در صورت عدم تشخیص و درمان سریع، می‌توانند منجر به مشکلات جدی و حتی مرگ ناگهانی شوند. تشخیص زودهنگام آریتمی از طریق ECG یک روش مؤثر برای پیشگیری از عوارض قلبی است. مدل پیشنهادی این پژوهه می‌تواند به دستگاه‌های مراقبت شخصی و مانیتورینگ پزشکی اضافه شود تا در صورت شناسایی آریتمی، بیمار سریعاً از شرایط خود آگاه شده و به پزشک مراجعه کند. چنین سیستمی می‌تواند به عنوان یک ابزار هشداردهنده برای افرادی که در معرض خطر بیماری‌های قلبی هستند، بسیار مفید باشد.

۲ معرفی مجموعه داده آریتمی MIT-BIH

پایگاه داده Arrhythmia IrMIT-BIH [۵] یکی از معتبرترین و پرکاربردترین منابع در حوزه تحلیل سیگنال‌های الکتروکاردیوگرام (ECG) و تشخیص آریتمی‌های قلبی است. این مجموعه داده در آزمایشگاه آریتمی بیمارستان بث اسرائیل در بوستون، آمریکا توسعه یافته و از سال ۱۹۸۰ در دسترس پژوهشگران قرار گرفته است. هدف اصلی این پایگاه داده، ایجاد یک استاندارد جهانی برای ارزیابی و توسعه الگوریتم‌های تحلیل ECG و تشخیص آریتمی بوده است.

۱.۲ مشخصات و ساختار مجموعه داده

پایگاه داده MIT-BIH شامل ۴۸ رکورد ECG است که هر رکورد حاوی ۳۰ دقیقه داده الکتروکاردیوگرام دو کanalه از ۴۷ بیمار واقعی است. این بیماران شامل نمونه‌هایی از افراد با ضربان نرمال و بیماران مبتلا به انواع مختلف آریتمی‌های قلبی هستند. داده‌ها از ۴۰۰۰ ثبت ۲۴ ساعته جمع‌آوری شده‌اند که از بین آن‌ها، ۲۳ مورد به صورت تصادفی انتخاب شده و ۲۵ مورد دیگر با هدف ثبت آریتمی‌های نادر و مهم بالینی در مجموعه قرار گرفته‌اند.

مشخصات فنی سیگنال‌های ECG:

- تعداد کanalهای دو کanal ECG در هر رکورد، شامل یک لید استاندارد و یک لید منتخب دیگر
- نرخ نمونه‌برداری: ۳۶۰ نمونه در ثانیه برای هر کanal، که دقت بالایی در تحلیل داده‌ها فراهم می‌کند
- روزولوشن دیجیتال: ۱۱ بیت در محدوده ۱۰ میلی‌ولت، به منظور تضمین کیفیت داده‌های ثبت شده
- مدت زمان هر رکورد: ۳۰ دقیقه، که شامل طیف گسترده‌ای از فعالیت‌های قلبی بیمار است
- فرمت‌های قابل دسترس: فرمت‌های MATLAB CSV WFDB و MATLAB که امکان تحلیل داده‌ها را در نرم‌افزارهای مختلف فراهم می‌کند
- حاشیه‌نویسی‌های تخصصی: بیش از ۱۱۰،۰۰۰ ضربان قلب با برچسب‌های دقیق که توسط متخصصان قلب بررسی و تایید شده‌اند
- کanal‌های ثبت شده: در هر رکورد، یک کanal مربوط به لید II Lead Modified (MLII) و کanal دوم یکی از لیدهای V1، V2، V6 یا V5 است. در برخی موارد، ترکیب لیدهای LL II Lead Limb (LL) و V1 نیز مشاهده می‌شود که بسته به بیمار متفاوت است.



۲.۲ حاشیه‌نویسی و برچسب‌گذاری

یکی از مهم‌ترین ویژگی‌های پایگاه داده MIT-BIH وجود حاشیه‌نویسی دقیق و برچسب‌گذاری تخصصی برای تشخیص آریتمی‌ها است. در این پایگاه داده:

- هر ضربان قلب به‌طور جداگانه برچسب‌گذاری شده و نوع آریتمی یا ضربان نرمال مشخص شده است.
- این برچسب‌گذاری‌ها توسط متخصصان قلب و عروق انجام شده و دقت بسیار بالایی دارند.
- اطلاعات مربوط به موقعیت زمانی (برحسب نمونه‌های ثبت شده) هر ضربان در فایل‌های متنه همراه با داده‌های سیگنال ثبت شده است.
- علاوه بر نوع ضربان، برخی فایل‌ها شامل اطلاعاتی درباره ویژگی‌های دیگر مانند سطح نویز، سیگنال‌های غیرقابل اعتماد، و رخدادهای خاص مرتبط با ریتم قلب هستند.
- داده‌های حاشیه‌نویسی در فایل‌های .atr و .txt ذخیره شده‌اند که شامل موقعیت‌های دقیق و نوع هر ضربان قلب است.

۳.۲ انواع آریتمی‌های ثبت شده

پایگاه داده MIT-BIH شامل انواع مختلفی از آریتمی‌های قلبی است که در پنج گروه اصلی دسته‌بندی شده‌اند:

جدول ۱: آریتمی‌های اصلی موجود در مجموعه داده

ضربان نرمال که در افراد سالم مشاهده می‌شود	beat Normal	N
ضربان زودرس دهلیزی که ناشی از فعالیت غیرطبیعی در دهلیز است	APB	A
انقباض زودرس بطئی که در اثر فعالیت ناگهانی در بطن‌ها رخ می‌دهد	PVC	V
همجوشی ضربان بطئی و نرمال که ترکیبی از ضربان طبیعی و یک ضربان آریتمی است	beat Fusion	F
ضربان نامشخص که در هیچ‌یک از دسته‌های استاندارد قرار نمی‌گیرد	beat Unclassifiable	Q

علاوه بر گروه‌های اصلی، این پایگاه داده شامل حاشیه‌نویسی‌های مرتبط با آریتمی‌های نادرتر نیز هست، مانند: سایر انواع آریتمی‌های ثبت شده

جدول ۲: سایر انواع آریتمی‌های ثبت شده در مجموعه داده

بلوک شاخه‌ای چپ که به علت انسداد در مسیر هدایت چپ قلب ایجاد می‌شود	LBBB	L
بلوک شاخه‌ای راست که اختلال در هدایت الکتریکی شاخه راست قلب را نشان می‌دهد	RBBB	R
ضربان زودرس گرهی که از گره AV منشا می‌گیرد	JPB	J
ضربان زودرس فوق‌بطئی که ناشی از فعالیت‌های غیرطبیعی در نواحی بالاتر از بطن است	SVPB	S
ضربان فراری بطئی که در موقع کاهش فعالیت نرمال گره SA رخ می‌دهد	VEB	E

این مجموعه داده به عنوان یک استاندارد بین‌المللی برای ارزیابی و توسعه الگوریتم‌های تحلیل ECG و تشخیص آریتمی شناخته شده است.



۳ پیش‌پردازش

۱.۳ بارگذاری مجموعه داده

برای آماده‌سازی مجموعه داده معرفی شده جهت آموزش مدل‌ها نیاز است دیتاست روی محیط Colab Google بارگیری شود. برای این کار فایل داده‌ای سیگنال و لیل‌های زمان رخداد هر آریتمی، به صورت یک فایل فشرده در وبسایت مجموعه داده قرار دارد، دانلود و به روی Drive Google از gdown به صورت زیر استفاده شد:

```
1 !gdown 1jndKYtmHxRq1I7wNeKXuEQ7t0WYukfyu -O mitbih_database.zip
2 !unzip mitbih_database.zip -d mitbih_database
```

بخش کد مربوط به بارگیری مجموعه داده: ۱ کد

در ادامه با استفاده از کد زیر تک فایل‌های موجود در مجموعه داده خوانده شدن و فایل‌های سیگنال از لیل جدا شدند.

```
1 # Separate records and annotation files
2 for name in filenames:
3     filename, file_extension = os.path.splitext(name)
4     if file_extension == '.csv':
5         records.append(path + filename + file_extension)
6     if file_extension == '.txt':
7         annotations.append(path + filename + file_extension)
```

بخش کد مربوط به جداسازی فایل‌ها: 2 کد

در ادامه یک نمونه از نوع فایل لیل‌ها آمده است.

Time	Sample #	Type	Sub Chan	Num	Aux
0:00.050	18	+	0	0	0 (N)
0:00.214	77	N	0	0	0
0:01.028	370	N	0	0	0
0:01.839	662	N	0	0	0
0:02.628	946	N	0	0	0
0:03.419	1231	N	0	0	0
0:04.208	1515	N	0	0	0
0:05.025	1809	N	0	0	0
0:05.678	2044	A	0	0	0

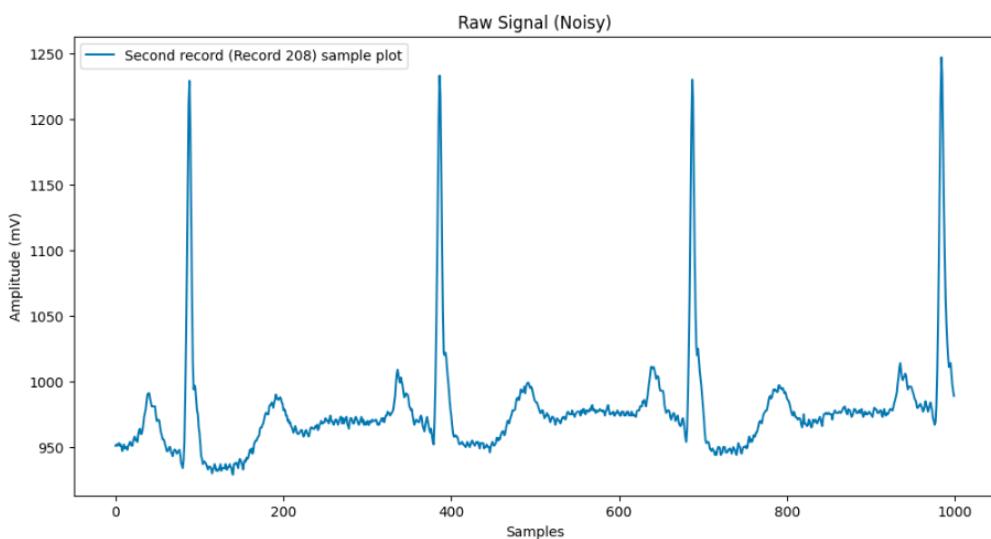
شکل ۲: نمونه‌ای از لیل‌ها



همانطور مشاهده می‌شود و در توضیحات مجموعه داده آمده است، لیل هر ضربان برای نمونه‌ای ثبت شده است که نشان دهنده نقطه R در آن ضربان است.

۲.۳ بهبود کیفیت سیگنال‌ها

در ادامه هزار سهیل از یکی از رکوردهای سیگنال از مجموعه داده رسم شده است.



شکل ۳: سیگنال خام

همانطور که مشاهده می‌شود، این سیگنال بسیار نویزی است و مقادیر دامنه آن برای مدل‌های یادگیری ماشین زیاد هستند. پس نیاز است چند مرحله برای بهبود کیفیت سیگنال و نرمال‌سازی دامنه آن انحلם شود. برای این کار از دو فیلتر Notch در فرکانس‌های ۳۰ و ۶۰ هرتز استفاده خواهد شد. علت انتخاب این فرکانس‌ها تداخل برق شهر روی دستگاه‌های ثبت سیگنال است. همچنین فیلتر دوم ۳۰ هرتزی هم به دلیل آن انتخاب شده است که در تعدادی از رکوردها، در فرآیند دیجیتالی کردن سیگنال‌های آنالوگ، سرعت ضبط دو برابر شده است و این تداخل برق شهر روی فرکانس ۳۰ هرتز قرار گرفته است. در انتهای هم یک فیلتر میانگین متحرک در نظر گرفته شد تا کیفیت سیگنال بهبود یابد. این فیلترها به صورت زیر پیاده‌سازی شدند:

```
1 def smooth_ecg(ecg_signal, window_size=10, fs=360):
2     kernel = np.ones(window_size) / window_size
3     denoised_signal = np.convolve(ecg_signal, kernel, mode='same')
4     return denoised_signal
5
6 def notch_filter60Hz_ecg(ecg_signal, notch_freq=60, fs=360, quality_factor=30)
7     :
8     # Design Notch filter
```



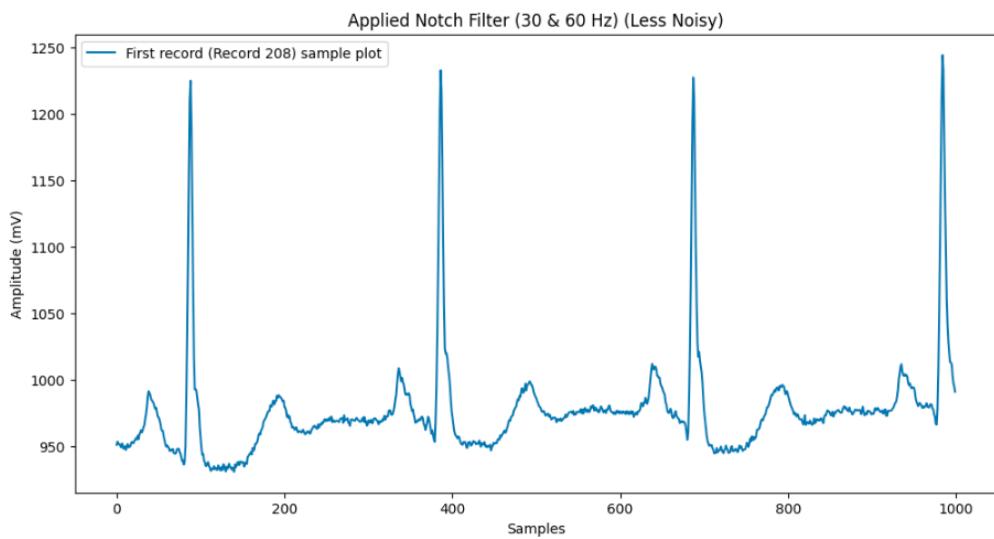
```

8     w0 = notch_freq / (fs / 2) # Normalize frequency
9     b, a = iirnotch(w0, quality_factor)
10    # Apply the Notch filter
11    filtered_signal = filtfilt(b, a, ecg_signal)
12    return filtered_signal
13
14 def notch_filter30Hz_ecg(ecg_signal, notch_freq=30, fs=360, quality_factor=30):
15     :
16     # Design Notch filter
17     w0 = notch_freq / (fs / 2) # Normalize frequency
18     b, a = iirnotch(w0, quality_factor)
19     # Apply the Notch filter
20     filtered_signal = filtfilt(b, a, ecg_signal)
21     return filtered_signal

```

فیلترها: ۳ کد

در ادامه نتیجه اعمال فیلترها در هر مرحله روی سیگنال آمده است:



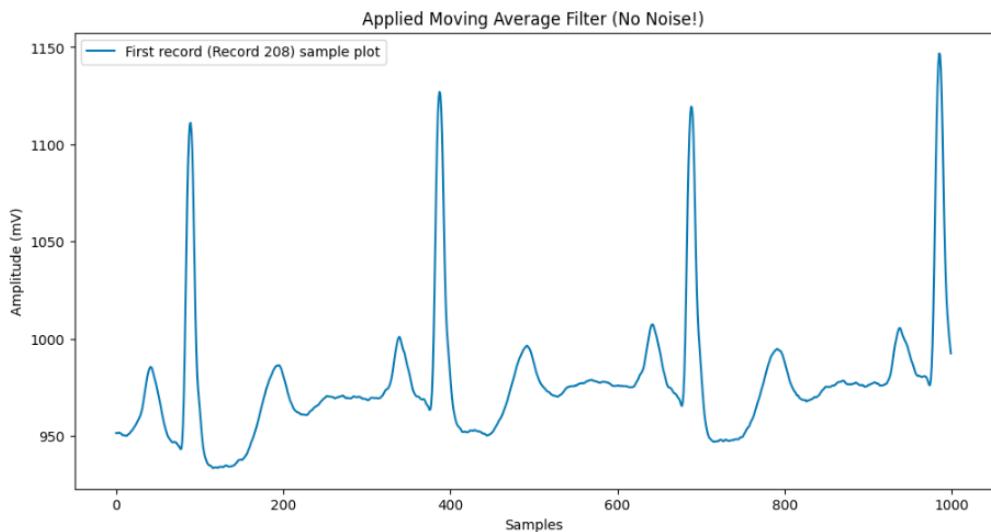
شکل ۴: سیگنال بعد از اعمال فیلتر Notch

در انتها نیز یک نرمال‌سازی به روش Z-Score انجام شد. تابع انجام این نرمال‌سازی در زیر آمده است:

```

1 def z_score_normalize(ecg_signal):
2     ecg_signal = np.array(ecg_signal)

```



شکل ۵: سیگنال بعد از اعمال فیلتر Average Moving و Notch

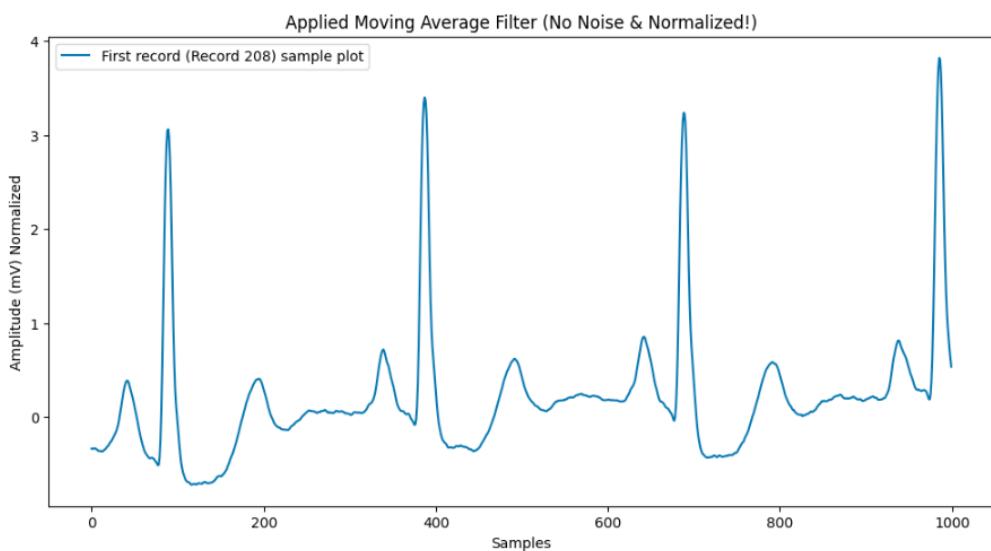
```

3 mean = np.mean(ecg_signal)
4 std = np.std(ecg_signal)
5 normalized_signal = (ecg_signal - mean) / std
6 return normalized_signal

```

نرمال‌سازی کد ۴: Z-Score

در نهایت سیگنال بهبود یافته به صورت زیر درآمد:



شکل ۶: سیگنال بهبود یافته نهایی



۳.۳ جداسازی تیشرها

برای این پروژه، هدف دستیابی آریتمی‌هایی است که در یک ضربان رخ می‌دهند، مانند آریتمی PVC. پس برای این نیاز است ضربان‌های موجود در همه رکوردها جدا شوند و با توجه به برچسبشان در لیست‌های X و y که ورودی و هدف‌های مورد استفاده در مدل‌ها خواهد بود، ذخیره شوند. کد زیر تمام فایل‌های جدا شده را باز می‌کند و پس از بهبود کیفیت سیگنال‌ها، تپش‌هایی که لیل‌های، L، A، V، N، F و R دارند را جدا و ذخیره می‌کند. لازم به ذکر است که در این بخش طول هر تپش ثابت و ۲۵۰ سمیل در نظر گرفته شد.

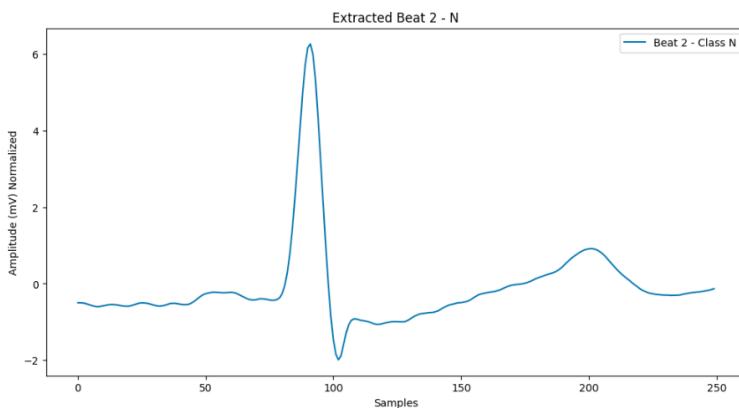


```
28
29     # Remove ECG signal Noise
30     signals = Remove_Noise(signals)
31     # Normalize signal using z-score
32     signals = z_score_normalize(signals)
33
34     # Extract beats based on annotations
35     with open(annotations[r], 'r') as fileID:
36         data = fileID.readlines()
37         for d in range(1, len(data)): # Skip header
38             splitted_annotation = data[d].split() # Split by whitespace
39             if len(splitted_annotation) < 3:
40                 continue # Skip invalid lines
41             pos = int(splitted_annotation[1]) # Extract Sample Index directly
42             arrhythmia_type = splitted_annotation[2] # Extract Arrhythmia
43             label
44
45             if arrhythmia_type in classes: # Check if beat type is relevant
46                 arrhythmia_index = classes.index(arrhythmia_type)
47                 count_classes[arrhythmia_index] += 1 # Count beats per class
48
49             if beat_size_afterR <= pos < (len(signals) - beat_size_afterR):
50                 :
51                 beat = signals[pos - beat_size_beforeR: pos +
52                               beat_size_afterR] # Extract beat
53                 X.append(beat)
54                 y.append(arrhythmia_index)
```

جداسازی تپش‌ها: ۵ کد



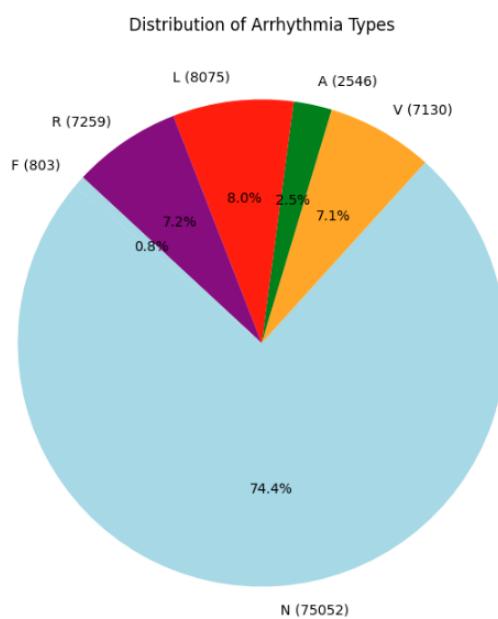
در ادامه یک نمونه از تپش‌های جدا شده با لیبل N آمده است:



شکل ۷: یک ضربان جدا شده

۴.۳ متعادل‌سازی مجموعه‌داده

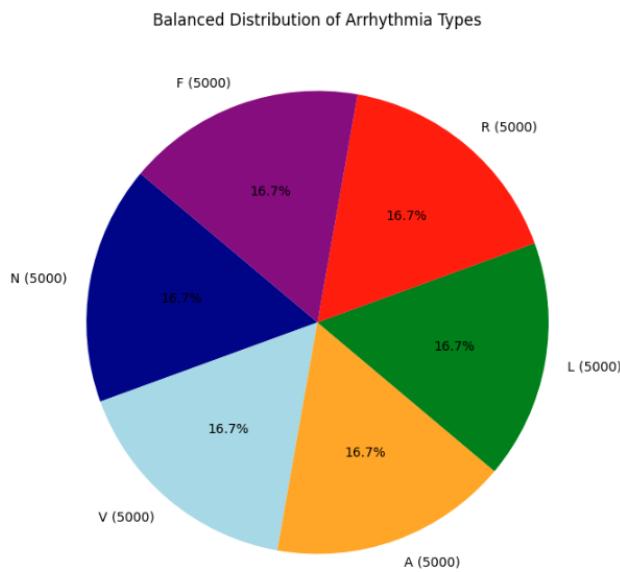
نمودار دایره‌ای تعداد هر لیبل (کلاس) جدا شده به صورت زیر است: همانطور که مشاهده می‌شود این تعداد متوازن نیست و این عدم



شکل ۸: توزیع تعداد اعضای هر کلاس

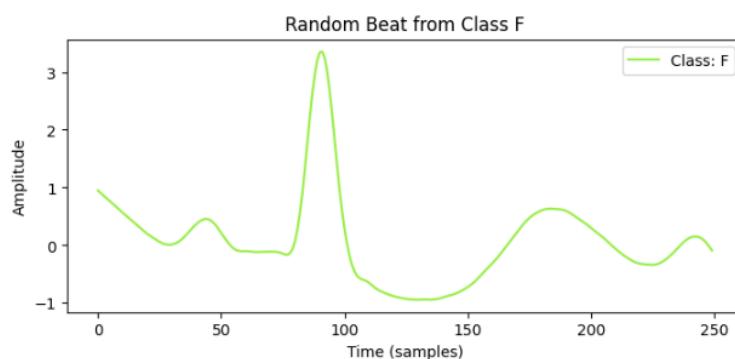
توازن موجب پیش آمدن بایاس در پیش‌بینی مدل‌ها خواهد شد. پس برای رفع این مشکل با Under و Sampling Over هر کدام از کلاس‌ها به ۵۰۰۰ رسیدند.

نمودار دایره‌ای تعداد هر لیبل (کلاس) بعد از متعادل‌سازی به صورت زیر درآمد:

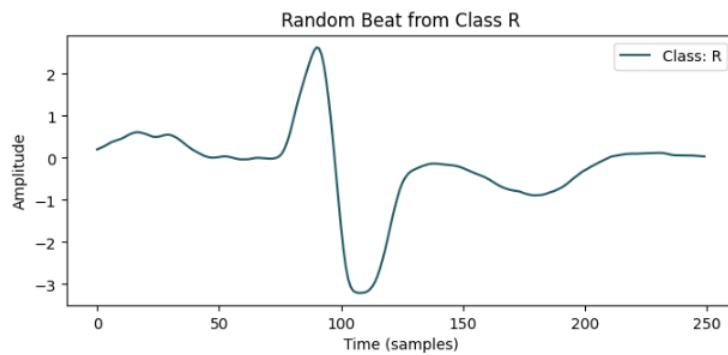


شکل ۹: توزیع تعداد اعضای هر کلاس بعد از متعادل‌سازی

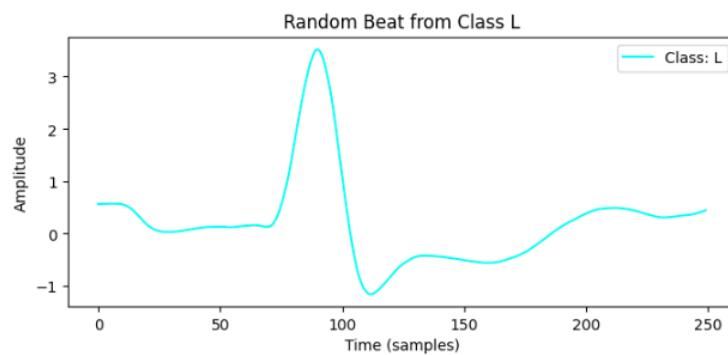
در انتها یک نمونه از ضربان در هر کلاس آمده است:



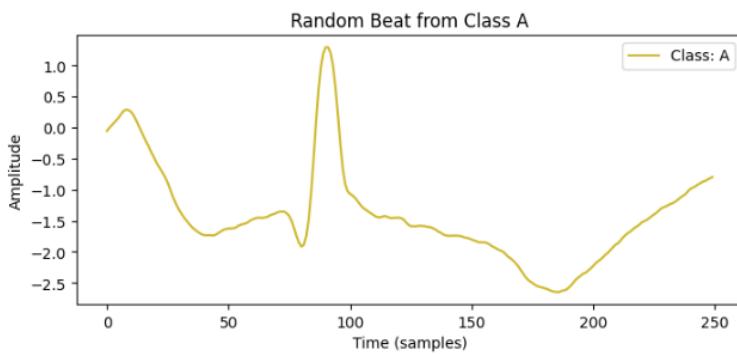
شکل ۱۰: نمونه ضربان کلاس F



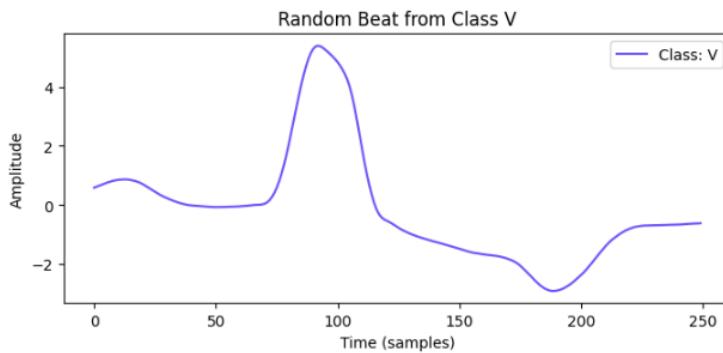
شکل ۱۱: نمونه ضربان کلاس R



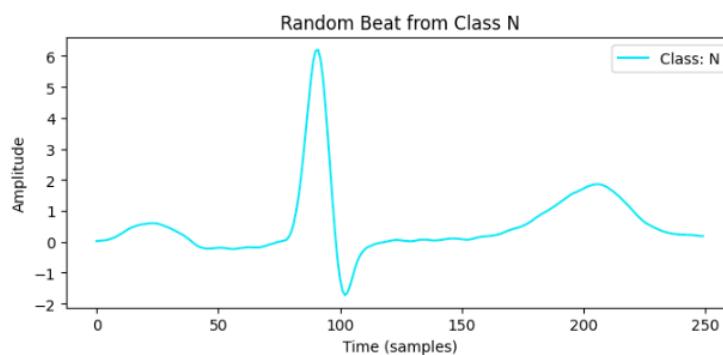
شکل ۱۲: نمونه ضربان کلاس L



شکل ۱۳: نمونه ضربان کلاس A



شکل ۱۴: نمونه ضربان کلاس V



شکل ۱۵: نمونه ضربان کلاس N

۴ مدل‌های پیاده‌سازی شده

۱.۴ مدل اول: رگرسیون لاجستیک Logistic Regression

۱.۱.۴ علت انتخاب مدل رگرسیون خطی و بررسی نقاط قوت و ضعف آن

رگرسیون لاجستیک یکی از ساده‌ترین و پرکاربردترین مدل‌های یادگیری ماشین است که برای دسته‌بندی و پیش‌بینی داده‌های عددی مورد استفاده قرار می‌گیرد. در این پژوهه، مدل رگرسیون لاجستیک انتخاب شد زیرا:

- سادگی و تفسیرپذیری بالا: این مدل به راحتی قابل پیاده‌سازی است و ضرایب وزن آن نشان‌دهنده تأثیر هر ویژگی در پیش‌بینی خروجی هستند.
- سرعت بالای آموزش: در مقایسه با مدل‌های پیچیده‌تر مانند شبکه‌های عصبی، این مدل از سرعت بسیار بالایی در آموزش و استنتاج برخوردار است.
- قابلیت تعمیم‌دهی: در صورت وجود رابطه‌ی خطی بین ویژگی‌ها و خروجی، مدل می‌تواند تعمیم خوبی به داده‌های جدید داشته باشد.

با این حال، رگرسیون خطی دارای محدودیت‌هایی نیز هست:



- عدم توانایی در مدل‌سازی روابط پیچیده: این مدل فرض می‌کند که رابطه‌ی بین متغیرهای ورودی و خروجی یک رابطه‌ی خطی است، در حالی که بسیاری از داده‌های پزشکی مانند ECG دارای الگوهای غیرخطی هستند.
- حساسیت به داده‌های پرت: وجود نویز یا مقادیر پرت می‌تواند به شدت بر عملکرد مدل تأثیر بگذارد.
- عدم کارایی در داده‌های چندبعدی با همبستگی پیچیده: در صورتی که ویژگی‌های ورودی به صورت غیرخطی وابسته باشند، مدل رگرسیون خطی عملکرد مطلوبی نخواهد داشت.

در این پژوهه، مدل رگرسیون خطی با هدف ارزیابی یک معیار پایه (Baseline) برای مقایسه با مدل‌های پیچیده‌تر مانند شبکه‌های عصبی عمیق پیاده‌سازی شد. این مدل می‌تواند به عنوان مرجع عملکرد پایه استفاده شود تا میزان بهبود سایر روش‌های یادگیری عمیق بررسی شود.

۲.۱.۴ نحوه پیاده‌سازی و توضیح کد مدل رگرسیون خطی

قبل از آموزش مدل، داده‌های ECG که شامل دنباله‌ای از ضربان‌های قلبی هستند، باید برای ورودی به مدل رگرسیون خطی آماده شوند. از آنجایی که رگرسیون خطی به داده‌های عددی ثابت (یک بعدی) نیاز دارد، ضربان‌های ECG که دارای مقدار دامنه بر حسب زمان هستند، به ویژگی‌های عددی برداری تبدیل شده‌اند. ابتدا سیگنال ECG که یک بردار زمانی است، به فرم دو بعدی تبدیل شده تا در مدل‌های یادگیری ماشین قابل استفاده باشد.

```
1 X_train_flat = X_train.reshape(X_train.shape[0], -1)
2 X_test_flat = X_test.reshape(X_test.shape[0], -1)
3 X_val_flat = X_val.reshape(X_val.shape[0], -1)
```

بخش کد مربوط به تبدیل داده‌ها به بردار عددی: 6 کد

سپس در ادامه این کد یک پایپ لاین یادگیری ماشین را ایجاد و آموزش می‌دهد که شامل نرمال‌سازی ویژگی‌ها با StandardScaler() و رگرسیون لجستیک با ۵۰۰۰ تکرار است:

```
1 # Logistic Regression Pipeline with Scaling
2 logistic_model = Pipeline([
3     ('scaler', StandardScaler()),    # Normalize input
4     ('classifier', LogisticRegression(max_iter=5000, random_state=13))
5 ])
6
7 # Train Model
8 logistic_model.fit(X_train_flat, y_train)
```

بخش کد مربوط تعریف مدل رگرسیون لجستیک: 7 کد



۳.۱.۴ نتایج مدل رگرسیون خطی

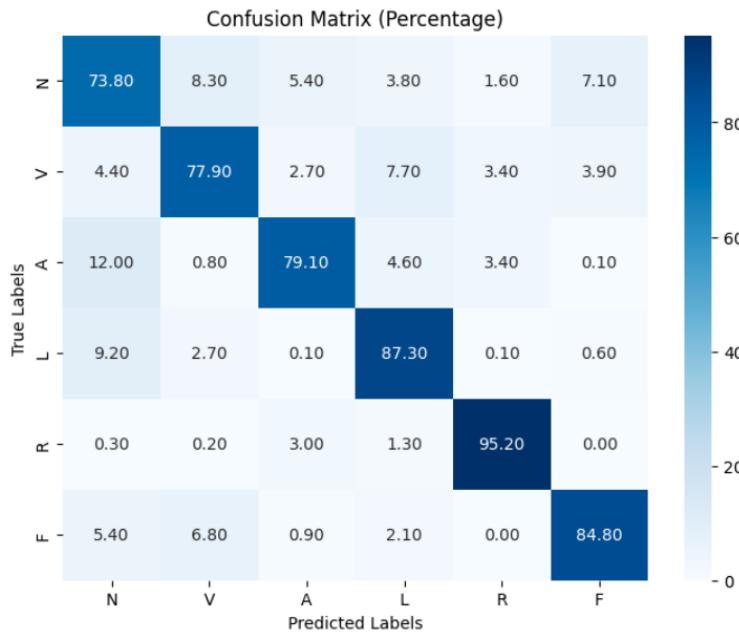
پس آموزش این مدل روی داده آماده شده از بخش قبل، نتایج زیر به دست آمدند:

	precision	recall	f1-score	support
N	0.70	0.74	0.72	1000
V	0.81	0.78	0.79	1000
A	0.87	0.79	0.83	1000
L	0.82	0.87	0.84	1000
R	0.92	0.95	0.93	1000
F	0.88	0.85	0.86	1000
accuracy			0.83	6000
macro avg	0.83	0.83	0.83	6000
weighted avg	0.83	0.83	0.83	6000

شکل ۱۶: گزارش طبقه‌بندی - مدل رگرسیون خطی

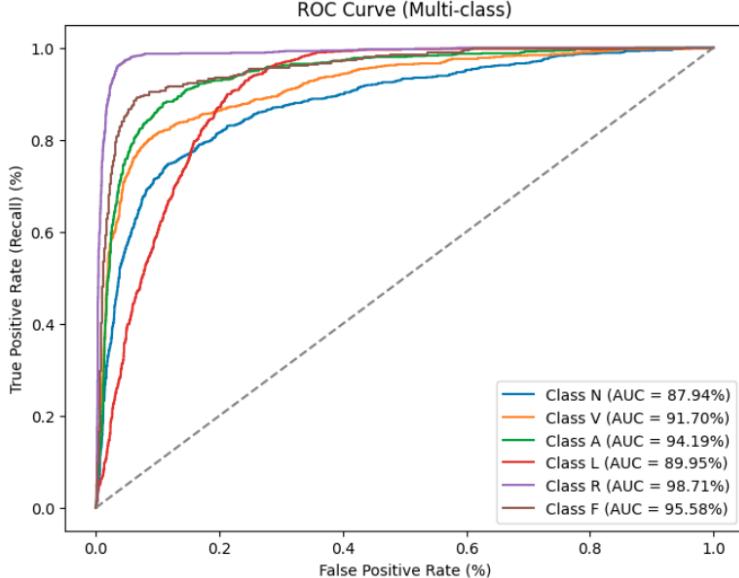
همانطور که مشاهده می‌شود دقت این مدل برای پیش‌بینی حدود ۸۳ درصد است. در ادامه ماتریس در هم آمیختگی این مدل زیر رسم

شد:



شکل ۱۷: ماتریس در هم آمیختگی - مدل رگرسیون خطی

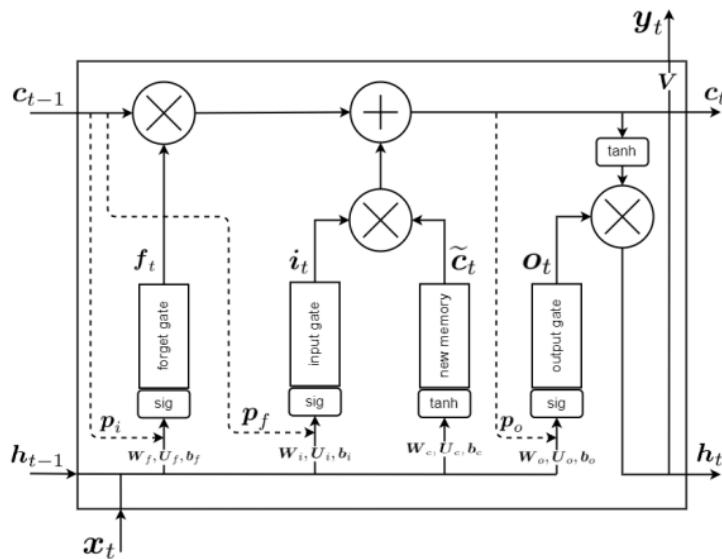
در این ماتریس مشاهده می‌شود که مدل رگرسیون لاجستیک در ضربان‌های نرمال و ضربان‌های دارای آریتمی PVC ضعیف‌ترین عملکرد را داشت. منحنی ROC مدل:



شکل ۱۸: نمودار ROC - مدل رگرسیون خطی

۲.۴ مدل دوم: Long Short-term Memory (LSTM)

شبکه‌های عصبی حافظه کوتاه‌مدت بلند LSTM - Memory Short-Term Long RNN - Networks Neural سنتی، با افزایش طول توالی، مشکل محو شدن گرادیان gradient vanishing رخ می‌دهد که یادگیری وابستگی‌های بلندمدت را دشوار می‌سازد. LSTM با معرفی ساختاری خاص شامل سلول‌های حافظه و مکانیزم‌های کنترلی، این مشکل را برطرف می‌کند و امکان یادگیری وابستگی‌های طولانی‌مدت را فراهم می‌نماید [۷].



[۷] ساختار یک سلول LSTM

هر واحد LSTM از سه دروازه تشکیل شده است: دروازه ورودی، دروازه خروجی و دروازه فراموشی. این دروازه‌ها با استفاده از توابع سیگموید، جریان اطلاعات را در داخل سلول کنترل می‌کنند. دروازه فراموشی تعیین می‌کند که چه مقدار از اطلاعات قبلی باید حفظ یا حذف شود؛ دروازه ورودی مشخص می‌کند که چه مقدار از اطلاعات جدید وارد سلول شود؛ و دروازه خروجی تصمیم می‌گیرد که چه مقدار از حالت سلول به خروجی منتقل گردد. این مکانیزم‌ها به LSTM امکان می‌دهند تا اطلاعات مهم را در طول توالی حفظ کرده و اطلاعات غیرضروری را حذف کنند [۷].

به دلیل این ساختار، LSTM در کاربردهایی مانند تشخیص گفتار، ترجمه ماشینی، تحلیل سری‌های زمانی و پردازش زبان طبیعی بسیار مؤثر است. توانایی LSTM در یادگیری وابستگی‌های بلندمدت و کوتاه‌مدت، آن را به ابزاری قدرتمند برای مدل‌سازی داده‌های ترتیبی تبدیل کرده است [۷]. در این بخش، مدل شبکه عصبی بازگشتی (LSTM) برای طبقه‌بندی سیگنال‌های ECG و تشخیص آریتمی‌های قلبی پیاده‌سازی شده است. استفاده از مدل‌های LSTM به دلیل توانایی آن‌ها در یادگیری وابستگی‌های زمانی در داده‌های سری زمانی مانند سیگنال‌های الکتروکاردیوگرام، انتخاب مناسبی برای این مسئله محسوب می‌شود.



۱.۲.۴ علت انتخاب مدل LSTM و بررسی نقاط قوت و ضعف آن

مزایای مدل LSTM:

- توانایی یادگیری توالی‌های بلندمدت: به دلیل داشتن سلول‌های حافظه، مدل LSTM می‌تواند ویژگی‌های بلندمدت وابسته به زمان را در سیگنال ECG به خوبی استخراج کند.
- عملکرد مناسب در داده‌های دارای نویز: به دلیل ساختار بازگشته، مدل LSTM می‌تواند الگوهای معنی‌دار را حتی در سیگنال‌های دارای نویز شناسایی کند.
- امکان پردازش سیگنال بدون استخراج دستی ویژگی‌ها: در مقایسه با روش‌های کلاسیک مانند SVM یا Logistic Regression، LSTM می‌تواند الگوهای معنی‌دار را حتی در سیگنال‌های دارای نویز شناسایی کند.
- دقت بالا در تشخیص آریتمی‌ها: در مقایسه با مدل‌های سنتی، LSTM معمولاً در دسته‌بندی داده‌های زیستی و پزشکی عملکرد بهتری دارد.

معایب مدل LSTM:

- نیاز به داده‌های زیاد: این مدل به حجم بالایی از داده‌های آموزشی نیاز دارد تا بتواند وابستگی‌های زمانی را به خوبی یاد بگیرد.
 - زمان آموزش بالا: به دلیل پیچیدگی محاسباتی بالاتر نسبت به روش‌های سنتی، آموزش مدل LSTM به زمان پردازش بیشتری نیاز دارد.
 - نیاز به پردازش قوی: مدل‌های مبتنی بر LSTM برای اجرا نیاز به واحد پردازش گرافیکی (GPU) یا سخت‌افزار قدرتمندتر دارند.
- با توجه به این ویژگی‌ها، LSTM یکی از مدل‌های مناسب برای تحلیل سیگنال‌های ECG و طبقه‌بندی آریتمی‌های قلبی است.

۲.۲.۴ نحوه پیاده‌سازی و توضیح کد مدل LSTM:

پیش‌پردازش داده‌ها: بعد از تعیین فرمت داده‌ها، شکل آن‌ها به صورت (تعداد نمونه‌ها، طول توالی، تعداد ویژگی‌ها) تغییر داده شد تا برای ورودی مدل آماده باشند. سپس لیبل‌های کلاس‌های مختلف به قالب Encoding One-Hot تبدیل می‌شوند تا در یادگیری مدل مؤثرتر باشند.

```
1 # Reshape data to fit LSTM (samples, timesteps, features)
2 X_train = X_train.reshape(X_train.shape[0], X_train.shape[1], 1)
3 X_val = X_val.reshape(X_val.shape[0], X_val.shape[1], 1)
4 X_test = X_test.reshape(X_test.shape[0], X_test.shape[1], 1)
5
6 # Convert labels to one-hot encoding
7 num_classes = 6 # 6 classes
8 y_train = to_categorical(y_train, num_classes)
9 y_val = to_categorical(y_val, num_classes)
```



```
10 y_test = to_categorical(y_test, num_classes)
```

بخش کد مربوط به آماده سازی داده‌ها برای مدل LSTM: ۸ کد

معماری مدل: ۱. ورودی مدل:

- سیگنال‌های ECG به عنوان یک توالی از نقاط نمونه‌برداری شده دریافت می‌شوند.
- اندازه ورودی به صورت (تعداد نمونه‌ها، طول توالی، تعداد ویژگی‌ها) تنظیم شده است.

۲. لایه‌های LSTM:

- یک لایه LSTM با ۶۴ واحد پردازشی برای استخراج ویژگی‌های سری زمانی از داده‌های ECG استفاده شده است.
- از `az sequences return = True` در لایه‌های ابتدایی استفاده شده تا خروجی هر لایه به لایه بعدی منتقل شود.
- یک لایه LSTM نهایی با ۳۲ واحد پردازشی برای یادگیری ویژگی‌های پیشرفته‌تر به مدل اضافه شده است.

۳. لایه‌های تمام‌متصل (Dense):

- یک لایه تمام‌متصل با ۶۴ نورون که الگوهای استخراج شده از لایه‌های LSTM را پردازش می‌کند.
- یک لایه خروجی (Softmax) با ۶ نورون (برای ۶ کلاس آریتمی) که احتمال هر کلاس را محاسبه می‌کند.
- یک لایه LSTM نهایی با ۳۲ واحد پردازشی برای یادگیری ویژگی‌های پیشرفته‌تر به مدل اضافه شده است.

۴. بهینه‌سازی مدل:

- تابع هزینه‌ی crossentropy categorical برای ارزیابی عملکرد مدل استفاده شده است.
- الگوریتم Adam به عنوان بهینه‌ساز مدل برای افزایش دقت و کاهش خطأ به کار رفته است.
- مدل برای ۴۰۰ دوره (Epoch) آموزش داده شده و در هر مرحله عملکرد آن بر روی داده‌های اعتبارسنجی ارزیابی شده است.

خلاصه معماری مدل: LSTM

در کد زیر معماری شرح داده شده پیاده‌سازی شد:

```
1 def build_lstm_model(input_shape, num_classes):
2     model = Sequential([
3         LSTM(64, return_sequences=True, input_shape=input_shape),
4         BatchNormalization(),
5         Dropout(0.3),
6
7         LSTM(32, return_sequences=False),
8         BatchNormalization(),
9         Dropout(0.3),
```



Layer (type)	Output Shape	Param #
lstm_2 (LSTM)	(None, 250, 64)	16,896
batch_normalization_2 (BatchNormalization)	(None, 250, 64)	256
dropout_3 (Dropout)	(None, 250, 64)	0
lstm_3 (LSTM)	(None, 32)	12,416
batch_normalization_3 (BatchNormalization)	(None, 32)	128
dropout_4 (Dropout)	(None, 32)	0
dense_2 (Dense)	(None, 32)	1,056
dropout_5 (Dropout)	(None, 32)	0
dense_3 (Dense)	(None, 6)	198

شکل ۲۰: معماری مدل LSTM استفاده شده

```

10
11     Dense(32, activation='relu'),
12     Dropout(0.3),
13     Dense(num_classes, activation='softmax')
14 )
15
16 model.compile(optimizer=Adam(learning_rate=0.001),
17                 loss='categorical_crossentropy',
18                 metrics=['accuracy'])
19
20 return model

```

بخش کد مربوط پیاده سازی مدل ۹: LSTM

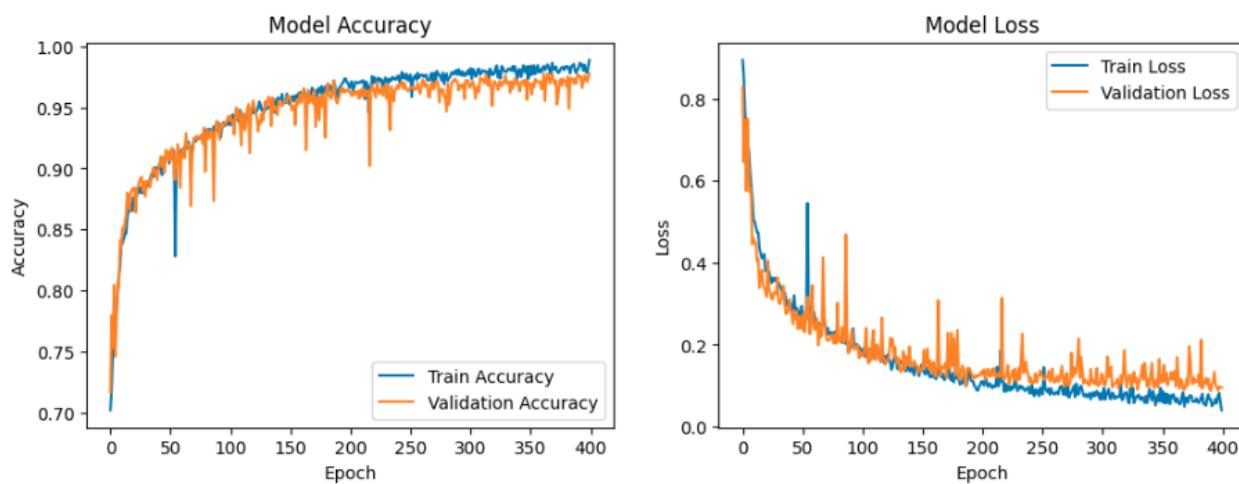
۳.۲.۴ نتایج مدل LSTM

پس آموزش این مدل روی داده آماده شده از بخش قبل، برای ۴۰۰ نتایج زیر حاصل شدند. نکته قابل توجه زمان خلی بیشتری است که آموزش مدل LSTM نیاز داشت. در ادامه گزارش طبقه‌بندی برای این مدل آمده است: همانطور که مشاهده می‌شود دقت این مدل برای پیشینی حدود ۹۸ درصد است.

Classification Report:				
	precision	recall	f1-score	support
0	0.97	0.96	0.96	1000
1	0.99	0.97	0.98	1000
2	0.97	0.97	0.97	1000
3	1.00	0.99	0.99	1000
4	0.99	0.99	0.99	1000
5	0.97	1.00	0.98	1000
accuracy			0.98	6000
macro avg	0.98	0.98	0.98	6000
weighted avg	0.98	0.98	0.98	6000

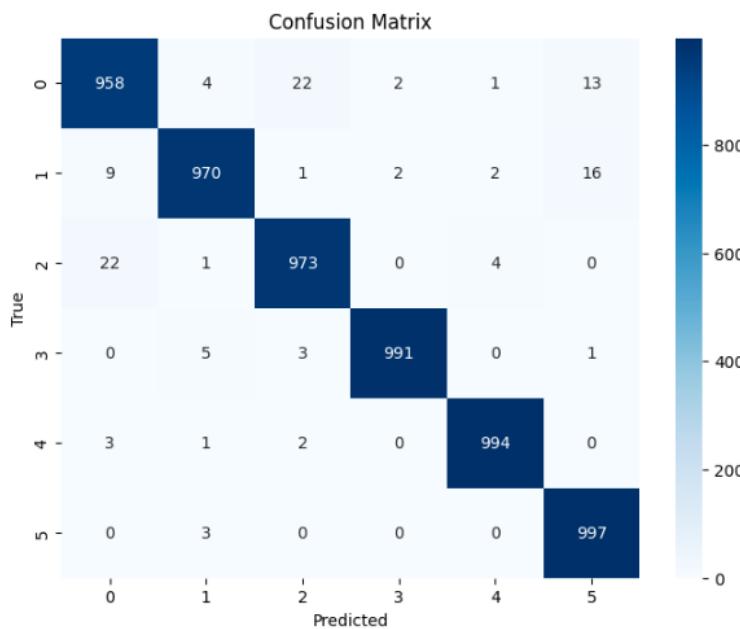
شکل ۲۱: گزارش طبقه‌بندی - مدل LSTM

در ادامه نمودار دقت و loss این مدل برای هر تکرار آمده است:



شکل ۲۲: نمودار Accuracy و Loss برای مدل LSTM

در ادامه ماتریس در هم آمیختگی این مدل زیر رسم شد. همانطور که انتظار می‌رفت عملکرد این مدل روی داده سیگنال ECG که یک سیگنال سری زمانی است، عملکرد بهتری داشت.



شکل ۲۳: ماتریس در هم آمیختگی - مدل LSTM

۵ نتیجه گیری

در نهایت، می‌توان نتیجه گرفت که برای تحلیل و طبقه‌بندی سیگنال‌های ECG استفاده از مدل‌هایی که قابلیت پردازش داده‌های سری زمانی را دارند، انتخاب بهتری خواهد بود. مدل‌های سنتی مانند رگرسیون لجستیک یا SVM، آگرچه در بسیاری از مسائل یادگیری ماشین کارآمد هستند، اما به دلیل عدم توانایی در یادگیری وابستگی‌های زمانی در سیگنال‌های فیزیولوژیکی، عملکرد مطلوبی روی داده‌های ECG نشان دهنده‌اند. در مقابل، مدل‌های شبکه‌های عصبی بازگشتی (RNNs) و بهخصوص LSTM که برای پردازش اطلاعات متوالی طراحی شده‌اند، توانایی قابل توجهی در تشخیص الگوهای مخفی در داده‌های ECG و طبقه‌بندی انواع آریتمی‌های قلبی دارند.

نتایج به دست آمده از آزمایش‌ها نشان داد که مدل LSTM عملکرد بسیار بهتری در تشخیص آریتمی‌های مربوط به مدل‌های کلاسیک داشت. این برتری ناشی از قابلیت نگهداری حافظه طولانی مدت در این مدل است که باعث می‌شود بتواند الگوهای پنهان در توالي سیگنال‌های ECG را شناسایی کند. با این حال، باید در نظر داشت که داده‌های مورد استفاده در این پژوهش تنها از یک مجموعه داده خاص استخراج شده‌اند و تمامی آزمایش‌ها درون مجموعه‌ای انجام شده‌اند. برای ارزیابی دقیق‌تر و افزایش قابلیت تعمیم مدل، ضروری است که این روش‌ها روی مجموعه داده‌های متنوع‌تر، شامل بیماران مختلف و شرایط واقعی‌تر، مورد ارزیابی قرار گیرند.

علاوه بر این، در کاربردهای بالینی، اجرای یک مدل باید علاوه بر دقت بالا، دارای سرعت پردازش مناسب، قابلیت تعمیم‌پذیری، و مقاومت در برابر نویز در داده‌های واقعی نیز باشد. مدل‌های پیچیده‌ای مانند CNN و LSTM به حجم داده‌های آموزشی بالا و توان پردازشی زیاد



نیاز دارند که ممکن است در محیط‌های عملیاتی واقعی (مانند دستگاه‌های پوشیدنی پزشکی یا سیستم‌های نظارت از راه دور) چالش‌هایی ایجاد کند. از این‌رو، در تحقیقات آتی، می‌توان با ترکیب مدل‌های سبک‌تر اما بهینه‌شده، مانند GRU یا CNN-LSTM، Hybrid LSTM به راه حل‌هایی دست یافت که هم از نظر دقیق‌تر تشخیص آریتمی‌ها و هم از نظر سرعت پردازش و بهره‌وری سخت‌افزاری کارآمد باشند. در مجموع، مدل LSTM به عنوان یک راهکار مؤثر در تحلیل سیگنال‌های ECG برای تشخیص آریتمی‌های قلبی شناخته می‌شود، اما برای کاربرد عملی آن در سیستم‌های مراقبت از سلامت و تجهیزات پزشکی، نیاز به بهینه‌سازی‌های بیشتری دارد که می‌تواند محور تحقیقات آینده باشد.



۶ لینک‌های مهم

کیتھاب (GitHub).

گوگل درایو (Google Drive)

گوگل کولب (Google Colab)



منابع

[۱] صفحه درس یادگیری ماشین.

- [2] G. N. Bhat, A. K. Ghatol, and A. S. Bhide, "ECG Signal Feature Extraction Trends in Methods and Applications," *BioMedical Eng. OnLine*, vol. 22, no. 1, pp. 1-25, 2023.
- [3] O. el B'Charri, L. Rachid, W. Jenkal, and A. Abenaou, "The ECG Signal Compression Using an Efficient Algorithm Based on the DWT," *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, pp. 1-7, Mar. 2016, doi: 10.14569/I-JACSA.2016.070325.
- [4] U. R. Acharya et al., "Automatic Detection of Arrhythmias Using Machine Learning and Deep Learning Algorithms," *Biomed. Signal Process. Control*, vol. 68, p. 102164, 2021, doi: 10.1016/j.bspc.2021.102164.
- [5] G. B. Moody and R. G. Mark, "The Impact of the MIT-BIH Arrhythmia Database," *IEEE Eng. Med. Biol. Mag.*, vol. 20, no. 3, pp. 45-50, 2001.
- [6] S. Khurshid, S. H. Choi, L.-C. Weng, E. Y. Wang, L. Trinquart, E. J. Benjamin, P. T. Ellinor, and S. A. Lubitz, "Frequency of Cardiac Rhythm Abnormalities in a Half Million Adults," *Circ. Arrhythm. Electrophysiol.*, vol. 11, no. 7, p. e006273, 2018.
- [7] B. Ghojogh and A. Ghodsi, "Recurrent Neural Networks and Long Short-Term Memory Networks: Tutorial and Survey," arXiv preprint arXiv:2304.11461, Apr. 2023.