Baseline em dataset de sinais fisiológicos coletados em mHealth

Artur de Vlieger Lima[1] **[13671574]**
Fernando Ferreira da Silva[1] **[13962425]**
Gabriel da Costa Merlin[1] **[12544420]**
Miller Matheus Lima Anacleto Rocha[1] **[13727954]**

[1]Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo
São Carlos – SP – Brazil

Resumo. *Esse artigo aborda o trabalho realizado pelos alunos para a disciplina SCC5977 - Aprendizado de Máquina para Séries Temporais ministrada pelo professor Diego Furtado Silva. Nesse trabalho, avaliamos o desempenho de diferentes métodos para séries temporais para tentar predizer condições de saúde de um indivíduo a partir de uma série temporal que representa o seu fluxo sanguíneo.*

1. Introdução

O monitoramento de sinais fisiológicos, sinais vitais e outros parâmetros que podem ser coletados ao longo do tempo são essenciais em diversas tarefas da área da Saúde, como a estimação de frequência cardíaca e a identificação de batimentos cardíacos anômalos. Essas séries temporais, no entanto, são obtidas por equipamentos muito custosos e usualmente não portáteis. Por outro lado, com a melhoria e miniaturização de sensores capazes de transmitir vários tipos de dados, dispositivos móveis e vestíveis têm se mostrado cada vez mais como opções para dar suporte a decisões médicas. *Smartphones* e *smartwatches* possuem sensores cada vez mais precisos e diversos, fazendo com que a Organização Mundial da Saúde considere que a chamada *mobile health* (mHealth) pode representar uma revolução no modo como populações interagem com sistemas públicos de saúde. Este trabalho se insere no contexto de um projeto que visa desenvolver técnicas para o aprendizado de modelos preditivos para esse tipo de dados.

2. Trabalhos relacionados

Com o desenvolvimento de novas tecnologias como veículos autônomos, cidades inteligentes e a Internet das Coisas, sensores móveis se tornaram cada vez mais baratos, precisos, pequenos e eficientes em uso de bateria e capacidade de processamento. Além disso, o barateamento de meios de armazenamento e o surgimento de algoritmos capazes de processar grandes volumes de dados permite, cada vez mais, a coleta de dados medidos ao longo do tempo em variados domínios do conhecimento, como saúde, economia e monitoramento ambiental. Como resultado, aplicações que coletam grandes volumes de dados temporais usando sensores surgem em diferentes domínios do conhecimento [Yeh 2018].

Especialmente, a área da Saúde conta com variadas formas de obtenção de dados temporais, bem como diferentes aplicações. Por exemplo, sinais relativos à atividade cardiorrespiratória podem auxiliar na prevenção de ataques cardíacos [Alsuliman and Sliman 2020] e sinais de unidades de medida inercial, como acelerômetros, pode auxiliar na identificação de doenças como o mal de Parkinson [El Maachi and Bouachir 2020].

3. Coleta de dados

Como a pesquisa pretende estimar parâmetros clínicos (*targets*) de uma pessoa, tal qual a quantidade de hemoglobina e leucócitos no sangue, por meio de séries temporais, é necessário a criação de um *dataset* contendo, para cada paciente, os valores de seus exames clínicos juntamente com uma série coletada que represente o fluxo sanguíneo em seu dedo.

Para a montagem do *dataset* foi necessário ir em postos de saúde que ofereciam o serviço de coleta de sangue, para que, caso um paciente aceitasse, o acesso aos resultados de seus exames seriam liberados e a coleta de sua série temporal seria feita seguindo o pipeline exposto a seguir:

1. Gravar um vídeo de 36s da ponta do dedo do paciente.
2. Cortar os primeiros e últimos 3s de vídeo a fim de evitar pertubações inicias e finais. Como o vídeo é feito em 30 FPS, temos 900 *frames* no vídeo final.
3. Dividir o vídeo em suas 3 componentes de cores: Red, Green, Blue.
4. Utilizar a média de cada frame para produzir uma observação da série temporal.
5. Z-normalizar as séries obtidas.

Dessa forma, teremos 3 séries temporais normalizadas produzidas, cada uma com 900 observações. Como todas elas referem-se a um mesmo paciente, teremos um dado multidimensional.

O dataset criado possui, até o momento, um total de 58 pacientes cadastrados contendo os dados de 78 exames clínicos. Caso alguma pessoa não tenha realizado um exame em específico, seu resultado será guardado como NaN. Além disso, a fim de proteger a privacidade dos pacientes que aceitaram participar da pesquisa, todos os dados foram anonimizados.

Um detalhe importante a ser citado é que há muito ruído nas séries temporais coletadas a partir de dispositivos móveis, que, por serem máquinas de propósito geral, não possuem maneira de estabilizar o movimento involuntário tanto dos pacientes quanto dos examinadores. Isso, somado à pouca quantidade de dados torna esse problema ainda mais desafiador para os modelos.

4. Metodologia

Com o *dataset* em mãos, foram selecionados 4 *targets* dos 78 presentes para serem preditos. Tendo em vista a relativa falta de dados (apenas 58 pacientes, muitos desses com exames faltantes), optou-se pelos *targets* que possuíam menos dados faltantes, sobrando eritrócitos, hemoglobina, leucócitos e plaquetas (todos com apenas 5 dados faltantes, logo temos 53 pacientes). Devido à falta de dados, o uso de algoritmos para *data augmentation* foi cogitado, mas como provocaram grande perda no desempenho dos modelos (aumentando erros na casa de ordens de grandeza) sendo no fim descartados. Os resultados serão discutidos e avaliados na próxima seção.

Para a predição, foram utilizados tanto métodos de aprendizado de máquina tradicionais quanto de redes neurais, analisando o problema pela ótica de classificação (com classes definidas com base em parâmetros médicos) e de regressão. Todos os modelos utilizados contaram com o uso de *Cross Validation* para se ter uma visão do desempenho.

4.1. Regressão

Tanto para os modelos tradicionais quanto para os de rede neural, foi utilizado a técnica de *K-Fold Cross Validation* com 10 *folds*, atentando-se para que ambas as abordagens utilizassem os mesmos dados em cada um dos *folds*, a fim de evitar que os resultados de uma delas fossem beneficiados em virtude de uma divisão mais fácil ou difícil.

Para a utilização do aprendizado de máquina tradicional, primeiro foi usado o *catch22* [Carl H Lubba 2019], um método capaz de inferir propriedades interpretáveis e com alto valor classificatório das séries temporais, para extrair *features* das séries temporais, e assim aplicarmos 4 modelos nessas features, sendo eles o *K-Nearest Neighbours* devido à sua simplicidade, o *Random Forest*, o *XGBboost* e o *CatBoost* por serem modelos de ensemble e assim possivelmente trazerem melhores resultados.

Já para as redes neurais não há pré-processamento, pois as séries já se encontram z-normalizadas no *dataset*, e também foram utilizados 4 modelos, *FCN* [WANG 2017], um modelo que inspirado no processamento de imagem, aplica convoluções nas séries temporais para tentar manter as relações temporais entre os dados, *MiniROCKET* [Angus Dempster 2020], um modelo que aplica várias convoluções fixas, aplicando um *ridge classifier* no final, *InceptionTime* [ISMAIL FAWAZ 2020], um ensemble do mesmo modelo executado 5 vezes, esse modelo apresenta blocos de Inception (várias convoluções de tamanhos diferentes), mas também conexão residual para evitar o *residual gradient* e assim conseguir ter mais camadas, e o *ResNet* [WANG 2017] que também consiste em camadas de convoluções com conexão residual, mas as convoluções não são em blocos como no Inception.

Ademais, é importante falar que nas redes neurais, além de já dividir com o *K-Fold* os dados em treino e teste, também usamos uma parte deles para validação. Portanto, os 53 pacientes que possuímos ficam divididos em 10% para treino, 10% para validação e 80% para teste. Por fim, destaca-se que cada experimento foi repetido 5 vezes a fim de diminuir o efeito da aleatoriedade dos pesos de cada neurônio.

4.2. Classificação

Para transformar o problema inicial num problema de classificação utilizamos os valores normais das variáveis *target* fornecidos em exames médicos, de forma que todos os valores dentro do intervalo esperado fossem rotulados como da classe 1, caso contrário seriam da classe 0. Os intervalos de esperados são mostrados abaixo.

$$12,0 \leq Hemoglobina[g/dL] \leq 16,0 \tag{1}$$

$$4.000 \leq Leucocitos[n^o/mm^3] \leq 10.000 \tag{2}$$

$$150.000 \leq Plaquetas[n^o/mm^3] \leq 450.000 \tag{3}$$

$$3,80 \leq Eritrocitos[x10^6/\mu L] \leq 5,00 \tag{4}$$

Analisando a distribuição de classes mostrada na Figura 1 percebemos que há um desbalanceamento entre as classes, a maioria das amostras está com valores dentro dos normais para cara *target*, um desbalanço neste nível quando se há poucas classes acaba afetando muito o treinamento do modelo, pois não temos exemplos o suficiente de ambas

as classes para dividi-los entre treino, validação e teste. Por este motivo não fizemos a divisão em *folds* como na regressão, pois não haveria como garantir que cada fold tivesse pelo menos um exemplo de cada classe.

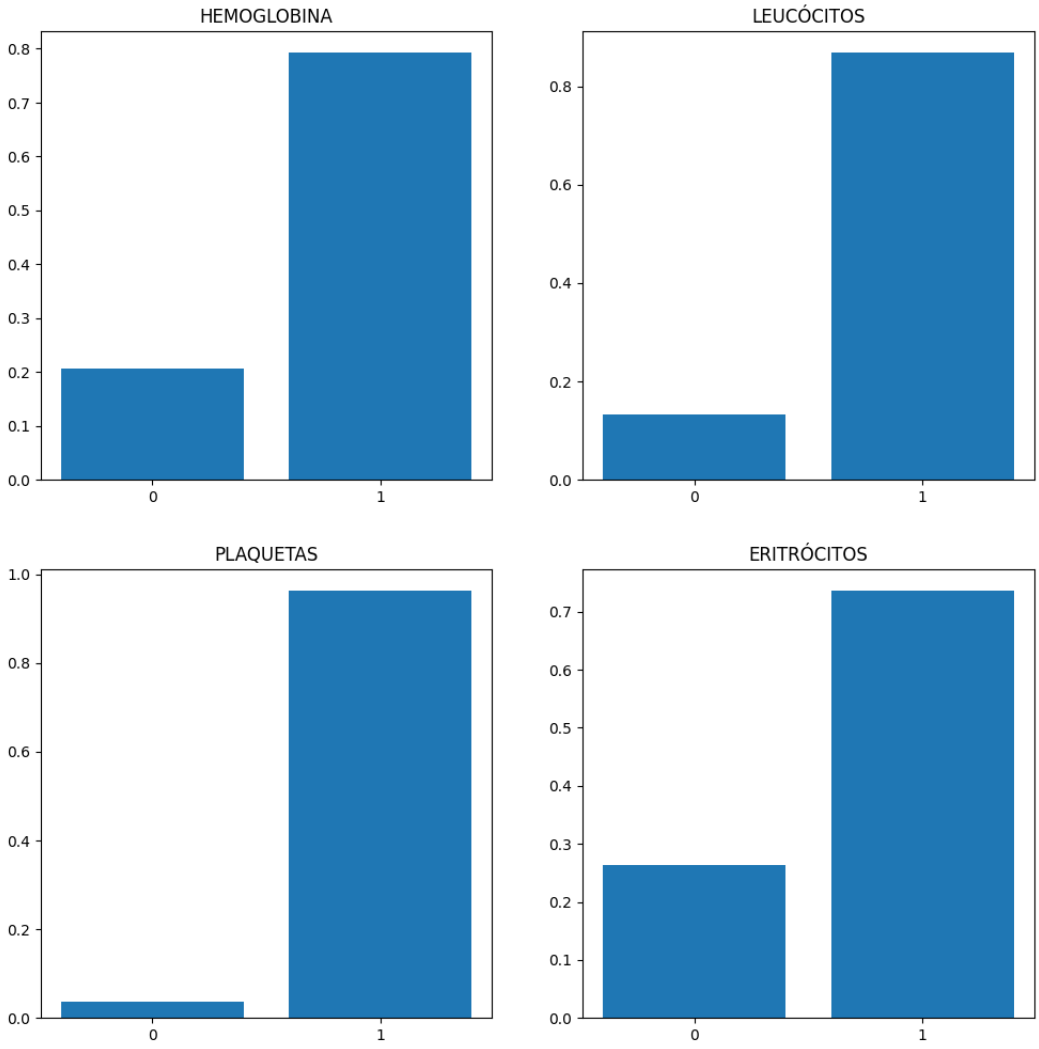

Figure 1. Distribuição das classes geradas a partir dos valores normais dos targets.

Assim como na regressão, utilizamos quatro *catch22* [Carl H Lubba 2019] aliado a modelos clássicos de aprendizado de máquina e quatro modelos *Deep Learning* para séries temporais, os modelos utilizados foram *K-Nearest Neighbours*, *Random Forest*, *XGBboost* e *CatBoost*, *FCN*, *ROCKET*, *InceptionTime* e *ResNet*.

Vale também informar que separamos 10% do *dataset* para testes em todos os teste e nos modelos de *Deep learning* usamos 12% para validação de forma a tentar garantir tenhamos pelo menos um exemplo de cada classe em cada subconjunto, isso nem sempre foi possível como no caso das plaquetas que não tem exemplos o suficiente da classe 0 para todos os subconjuntos. Além disso fizemos cada teste 25 vezes para termos resultados mais próximos dos reais e menos dependentes da aleatoriedade.

5. Resultados

Abaixo estão tabelas contendo os valores dos resultados obtidos a partir do treinamento de cada um dos modelos, levando em conta cada *target*. No caso da Classificação, tem-se os números do F1 e da acurácia. Enquanto que para a Regressão, escolheu-se expor os valores do Erro Médio Absoluto (MAE - *Mean Absolute Error*) e do Erro Médio Quadrado (MSE - *Mean Squared Error*), pois, enquanto o MSE penaliza mais os modelos que não foram bem, o que facilita a observação dos resultados, o MAE é uma métrica que traz muito mais valor semântico ao estudo, já que, apenas comparando com o valor normal dos exames clínicos, pode-se saber se o modelo treinado está com um erro muito grande ou baixo em relação ao valor esperado.

5.1. Regressão

	Hemoglobina	**Leucócitos**	**Plaquetas**	**Eritrócitos**
Inception Time	1.492	1728.582	68861.910	0.468
ResNet	1.372	2008.336	64853.948	0.476
FCN	1.230	1753.066	60304.246	0.442
MINIROCKET	2.852	3919.352	122201.118	1.170
KNN	1.400	1596.533	**55563.127**	0.455
Random Forest	1.243	1817.673	60359.716	0.460
XGBoost	1.346	1572.274	57369.721	**0.436**
CatBoost	**1.189**	**1532.012**	56149.463	0.450

Table 1. Tabela de Erro Absoluto dos modelos em cada target (Regressão).

	Hemoglobina	**Leucócitos**	**Plaquetas**	**Eritrócitos**
Inception Time	3.356	4819498.590	$6.87e9$	0.379
ResNet	2.986	6181861.670	$6.78e9$	0.397
FCN	**2.376**	5325496.837	$5.93e9$	0.360
MINIROCKET	10.772	20256720.637	$2.11e10$	1.871
KNN	2.915	4942359.088	$4.87e9$	0.314
Random Forest	2.386	5954431.630	$5.60e9$	0.310
XGBoost	2.813	4423277.638	$5.46e9$	**0.298**
CatBoost	2.428	**4272569.927**	$5.09e9$	0.326

Table 2. Tabela de Erro Quadrado dos modelos em cada target (Regressão).

Pode ser visto, a partir das Tabelas 1 e 2, que os resultados obtidos a partir da extração de *features* possuem resultados um pouco menores, portanto melhores, para todos os *targets* analisados. Sendo que, entre cada um desses modelos não houve uma diferença expressiva. Por outro lado, entre as redes neurais houve somente diferença significativa entre os erros do MiniROCKET e das demais redes, algo que pode ter sido causado pela pequena quantidade de dados, que tornou difícil para os *kernels* fixos do MiniRocket a tarefa de Regressão. O aumento marginal dos erros das redes neurais, quando comparados com os outros modelos, também pode ser devido aos poucos dados,

já que, diferente do modelos de aprendizado de máquina clássicos, as redes neurais dependem de uma quantidade muito maior de dados para que sejam treinadas com um resultado satisfatório.

Por fim, precisa-se fazer uma análise semântica dos erros, ou seja, ponderar se os resultados obtidos são adequados para a tarefa proposta. Como se pode ver na seção anterior, os valores normais para o exames de Hemoglobina, Leucócitos, Plaquetas e Eritrócitos são, respectivamente, na casa de (em unidades padrão de cada medida): 14, 7000, 300000 e 4.40. Enquanto que os valores médios de erro absoluto obtidos pelos modelos, tanto de redes neurais (exceto MiniROCKET), quanto de extração de *features*, são: 1.32, 1715.50, 60494.60 e 0.46. Como se pode concluir, os erros obtidos estão entre 10% e 20%, um erro percentual que não pode ser admitido numa pesquisa sobre a saúde de pessoas, já que um desvio desse tamanho pode, facilmente, classificar uma pessoa sadia como doente e vice-versa.

5.2. Classificação

	Hemoglobina	**Leucócitos**	**Plaquetas**	**Eritrócitos**
Inception Time	0.403	**0.601**	0.945	0.527
ResNet	0.448	0.550	0.945	0.534
FCN	0.522	0.555	0.945	0.493
ROCKET	0.398	0.535	0.945	0.544
KNN	0.543	0.518	0.847	0.350
Random Forest	0.598	0.542	0.847	**0.563**
XGBoost	**0.641**	0.505	0.847	0.452
CatBoost	0.623	0.531	0.847	0.465

Table 3. Tabela de F1 dos modelos em cada target (Classificação).

	Hemoglobina	**Leucócitos**	**Plaquetas**	**Eritrócitos**
Inception Time	0.608	0.833	0.983	0.691
ResNet	0.658	0.825	0.983	0.700
FCN	0.749	**0.841**	0.983	0.708
ROCKET	0.616	0.808	0.983	0.700
KNN	0.760	0.793	0.953	0.540
Random Forest	0.800	0.806	0.953	**0.753**
XGBoost	**0.813**	0.766	0.953	0.646
CatBoost	0.793	0.780	0.953	0.700

Table 4. Tabela de Acurácia dos modelos em cada target (Classificação).

Já na classificação cada modelo foi treinado 25 vezes e foi feita a média tanto da acurácia quando to F1 macro. Os resultados apresentados nas Tabelas 3 e 4 mostram que em geral os modelos, como o *XGBosst* e o *CatBoost*, que trabalharam em cima das extração de *features* do *catch22* tem uma melhor performance na tarefa.

Também percebemos valores muito altos para acurácia das Plaquetas além de acurácias idênticas tanto entre os modelos que tem como entrada a série em si quanto

para os que usaram as *features* do *catch22*. Como já discutido na Seção 4, por termos uma quantidade muito limitada de dados ainda temos classes extremamente desbalanceadas. Isso sugere que todos os modelos estão nos retornando como resposta apenas a uma única classe que aparece mais na distribuição.

6. Conclusão

Com esse estudo, foi possível criar *baselines* para análise de um novo *dataset* de sinais fisiológicos coletados por meio de um *smartphone* utilizando para isso extração de *features* juntamente com aprendizado de máquina clássico e redes neurais. No entanto, os resultados conseguidos não foram bons, devido à diversos fatores já discutidos, sendo o principal deles a falta de dados.

Conforme dito, a pouca quantidade de dados é uma característica do *dataset* que dificulta demais o bom funcionamento dos modelos. Portanto, o principal trabalho que deve ser feito é a obtenção de mais dados.

No entanto, como o aumento do *dataset* é algo demorado, o estudo aprofundado de técnicas de *Data Augmentation* que se comportem bem para o domínio dos dados estudados também é uma possível solução para melhorar os resultados obtidos.

Por fim, como a coleta de séries temporais por meio da mHealth traz consigo diversos ruídos, o estudo de técnicas para extração deles trará uma melhora no desempenho dos modelos, já que, após seu uso, espera-se que apenas as informações importantes restarão nas séries temporais coletadas.

7. Códigos utilizados

Link para Github: https://github.com/gabrielcmerlin/testesDatasetMHealth

References

[Alsuliman and Sliman 2020] Alsuliman, T., H. D. and Sliman, L. (2020). Machine learning and artificial intelligence in the service of medicine: Necessity or potentiality?

[Angus Dempster 2020] Angus Dempster, Daniel F. Schmidt, G. I. W. (2020). Minirocket: A very fast (almost) deterministic transform for time series classification.

[Carl H Lubba 2019] Carl H Lubba, Sarab S Sethi, P. K. S. R. S. B. D. F. N. S. J. (2019). Catch22: Canonical time-series characteristics.

[El Maachi and Bouachir 2020] El Maachi, I., B. G.-A. and Bouachir, W. (2020). Deep 1d-convnet for accurate parkinson disease detection and severity prediction from gait.

[ISMAIL FAWAZ 2020] ISMAIL FAWAZ, H. e. a. (2020). Inceptiontime: Finding alexnet for time series classification. data mining and knowledge discovery.

[WANG 2017] WANG, Zhiguang; YAN, W. O.-T. (2017). Time series classification from scratch with deep neural networks: A strong baseline.

[Yeh 2018] Yeh, C.-C. M., K.-E. e. a. (2018). Time series joins, motifs, discords and shapelets: a unifying view that exploits the matrix profile.