PIBIC/CNPq - PIBTI - UFSC

Programa de Iniciação Científica
Relatório Final

Título do Projeto do Orientador	Otimização de Etapas do Fluxo de Síntese de Circuitos Integrados Explorando Técnicas de Aprendizado de Máquina
Título do Plano de Atividades do Bolsista	Predição da caracterização elétrica de circuitos com algoritmos de Machine Learning
Nome do Bolsista	Gabriel Lima Jacinto
Nome do Orientador	Cristina Meinhardt
Grupo de Pesquisa	Automação do Projeto de Sistemas Computacionais Embarcados - UFSC
Palavras-chave	nanotecnologia, projeto de circuitos integrados, energia, atraso, caracterização elétrica, machine learning, variabilidade
Período de Vigência da Bolsa	Setembro/2021 a Agosto/2022

Resumo

Com o avanço da tecnologia de fabricação de circuitos integrados, cada vez mais aspectos devem ser considerados durante a caracterização elétrica de circuitos a fim de solucionar desafios como os de efeito de variabilidade do processo. Isso aumenta o tempo de caracterização devido as tradicionais técnicas baseadas em simulações elétricas exaustivas. A adoção de técnicas de aprendizado de máquina já auxilia o projeto digital em muitos níveis de abstração. Assim, o objetivo principal desta pesquisa é avaliar algoritmos de regressão de aprendizado de máquina como alternativa à simulação elétrica exaustiva no projeto de caracterização de células. Nesta etapa, foram considerados os algoritmos de regressão linear múltipla, regressão de vetores de suporte, árvores de decisão e floresta aleatória. Este relatório apresenta os resultados de uma porta NAND2 usando a tecnologia *bulk* CMOS. Especificamente, prever-se-á separadamente os valores da energia e os tempos de propagação desse circuito. Uma análise comparativa, juntamente com o tempo de inferência, é feita para cada variável dependente entre os modelos, a fim de entender qual é o melhor modelo de regressão para a tarefa. O algoritmo com a função de menor custo e menor tempo de inferência provou ser as Árvores de Decisão para todas as variáveis preditas.

Introdução

O constante avanço no processo de fabricação de transistores permite o aumento da capacidade de integração de funcionalidades em um mesmo projeto, com melhores resultados de desempenho devido aos transistores serem mais rápidos, consumirem menos energia e

ocuparem menor área. Entretanto, esta evolução também insere novos desafios, principalmente ao projetarmos circuitos integrados em tecnologias nanométricas. Um dos maiores desafios nas tecnologias em nodos nanométricos é a variabilidade do processo. A variabilidade do processo afeta o comportamento normal esperado das células, alterando o atraso e o consumo de energia observados em condições padrões. Juntamente com os efeitos de variabilidade de processo, soma-se o aumento da complexidade das regras de projeto. Consequentemente, as ferramentas de projeto de circuitos integrados precisam resolver problemas cada vez mais difíceis. Esse problema vem crescendo exponencialmente, e essas dificuldades refletem diretamente no custo de desenvolvimento de projetos de dispositivos eletrônicos (KAHNG, 2018).

Assim, a caracterização da célula passa a incluir vários pontos, além dos tradicionais pontos que observam as características elétricas considerando os dispositivos mais rápidos, mais lentos e em condições nominais. Para explorar uma grande variedade de pontos de caracterização possível em uma tecnologia alvo, tradicionalmente os projetistas adotam simulações elétricas exaustivas para todos pontos, observando o impacto nas métricas de atraso e consumo de energia. Assim, aumentar o número de simulações elétricas para cobrir esses pontos pode se tornar uma tarefa de demanda de grande tempo.

Técnicas de aprendizado de máquina podem ser boas alternativas na tentativa de reduzir o tempo de caracterização e os custos para o desenvolvimento de dispositivos eletrônicos. A revisão bibliográfica mostra que ainda há muito espaço a ser explorado quanto ao uso de aprendizado de máquina para ferramentas na área da microeletrônica. Seu uso atual está fortemente ligado à previsão de qualidade de soluções candidatas dadas por ferramentas existentes, ou tentando guiá-las para uma solução de melhor qualidade, por meio de previsões métricas (KAHNG, 2018).

Com a disponibilidade crescente de grandes quantidades de dados de manufatura e as de melhoria de processo, técnicas de aprendizado de máquina podem ser adotadas para analisar, classificar e prever a qualidade da durante a etapa de manufatura que realiza a gravura dos metais no circuito (Chatterjee et al., 1996) ou investigar os efeitos da variabilidade do processo em células de memória *flash* do tipo 3DNAND (Lee; Ko; Shin, 2020). Nas etapas de posicionamento e roteamento de circuitos, encontramos trabalhos que usam aprendizado de máquina para prever alterações na caracterização de atrasos dos circuitos baseada em caminho de análise de tempo adotando gráficos (Kahng; Mallappa; Saul, 2018) e preditores de tempo de pré-roteamento (Barboza et al., 2019).

Além disso, o trabalho (Kaintura et al., 2018) explora aprendizado de máquina na caracterização elétrica de dispositivos, no entanto, especificamente para dispositivos magnéticos. Assim, salienta-se que há espaço para explorar também algoritmos de aprendizado de máquina para caracterização elétrica de células lógicas em tecnologias de *bulk* CMOS (*Complementary Metal Oxide Semiconductor*), SOI (*Silicon On Insulator*) ou FinFET (*Fin Field-Effect*).

A caracterização de confiabilidade de células combinacionais e sequenciais é tradicionalmente baseada em simulações elétricas exaustivas, elevando o tempo e custo de projeto.

Observando a crescente adoção das técnicas de aprendizado de máquina na área de microeletrônica, este projeto inicia uma nova linha de metodologia, buscando a aplicação destas técnicas como alternativa para substituir os tradicionais métodos de simulação de efeitos de falhas de radiação e de variabilidade de processo.

Desta forma, este trabalho tem como objetivo explorar técnicas de aprendizado de máquina para prever o comportamento elétrico de circuitos. Em trabalhos anteriores, considerou-se como um estudo de caso o circuito inversor CMOS, com o alvo de predição a energia e o tempo de propagação. Objetivando-se expandir e testar a técnica desenvolvida, este relatório explora mais uma porta CMOS: "não e"(NAND2). Levando em consideração os melhores modelos dos experimentos feitos com a porta inversora, optou-se por seguir com os quatro métodos métodos de regressão: Árvore de Decisão (DT), Floresta Aleatória (RF), Regressão Linear Múltipla (MLR) e Regressão de Vetores de Suporte (SVR). As variáveis de entrada consideradas foram a tensão, a capacitância, a temperatura e a variabilidade do processo. A metodologia de predição apresentada neste artigo será estendida para considerar outros circuitos e tecnologias nas etapas seguintes deste projeto.

Inicialmente nesse relatório discute-se sobre o referencial teórico básico dos algoritmos de aprendizado de máquina que foram utilizados, juntamente com a explicação dos cálculos matemáticos que são feitos nas predições dos modelos. Depois, explicam-se os métodos utilizados para se extrair os dados que treinaram os algoritmos, assim como as técnicas aplicadas em suas modelagens para evitar problemas de *overfit* e melhorar a performance dos modelos tomando como base a função de custo escolhida. Logo em seguida os resultados dos treinamentos são discutidos por meio tanto da análise dos erros da predições quanto pela representação gráfica desses erros. Apresentam-se também os resultados comparativos de tempo para averiguar se o métodos desenvolvido é realmente mais rápido do que as tradicionais simulações exaustivas. Por fim, sintetiza-se todo o processo realizado e os resultados obtidos em uma conclusão que indica qual foi o melhor algoritmo e o pior algoritmo para o objetivo posto nesse trabalho.

Algoritmos de Aprendizado de Máquina

Existem vários algoritmos de aprendizado de máquina aplicáveis para diferentes tarefas, que vão desde a previsão de valores no mercado de ações até a navegação de um robô em Marte. Uma vez que, neste trabalho, objetiva-se a previsão da energia e do atraso de um circuito não-e CMOS - que são todos valores contínuos - a abordagem será usar uma técnica de aprendizagem supervisionada conhecida como regressão, que calcula a correlação entre as variáveis de entrada (GÉRON, 2017). Existem muitos algoritmos de regressão, portanto, delimitou-se esta análise aos quatro mais comuns.

Regressão linear múltipla

A regressão linear é uma técnica estatística amplamente usada quando se deseja prever o comportamento de um valor alvo. Como neste trabalho as variáveis de saída são previstas com base nos valores de muitas variáveis de entrada, usou-se a regressão linear múltipla (MLR) em vez da regressão linear simples (Figura 1). O valor predito $\hat{y}$ é descrito na Eq. 1, onde θ_0 é o termo de interceptação e $\theta_1 ... \theta_n$ são os pesos de cada valor das variáveis de entrada (GÉRON, 2017). O que se deseja é encontrar os valores de cada θ_n que minimizem a função de custo do modelo (GÉRON, 2017), que neste trabalho será a raiz do erro quadrático médio (RSME).

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ... + \theta_n x_n \tag{1}$$

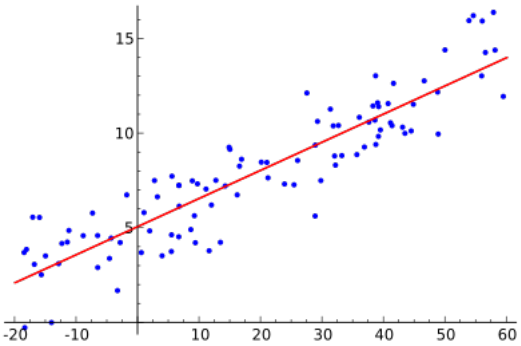

Figura 1: Visualização de Regressão Linear Simples

Regressão de vetores de suporte

Embora *Support Vector Machines* (SVM) sejam normalmente aplicados a problemas de classificação, é um algoritmo versátil que também pode realizar regressões, denominado SVR (GÉRON, 2017). A ideia é que, dado um valor ϵ, o SVR calcula uma margem em torno do hiperplano da função de previsão, onde tenta-se ajustar o máximo de observações possível entre a área definida pelas margens e reduzir os valores que são fora dessa mesma área (Fig. 2).

A função de decisão representada pela Eq. 3 é calculada de forma semelhante à regressão linear, entretanto, aqui optou-se por colocar a equação em uma forma vetorizada (GÉRON, 2017), que é a prática mais comum quando representamos algoritmos de aprendizado de máquina. Portanto temos que o valor predito é representado por $\hat{y}$, w^T é a matriz transposta de pesos para cada variável de entrada x (os valores $\theta_1 ... \theta_n$ da Eq. 1) e b é o valor de interceptação. O SVR tem muitos *kernels* (linear, RBF, polinomial) que são funções que capazes de realizar o produto escalar $\phi(a)^T \phi(b)$ apenas usando os valores de a e b, sem necessariamente calcular a transformação ϕ (GÉRON, 2017). Neste trabalho foi usado o *kernel* RBF representado pela Eq. 2.

$$K(a,b) = e^{-\gamma||a-b||^2} \tag{2}$$

$$\hat{y} = w^T x + b \tag{3}$$

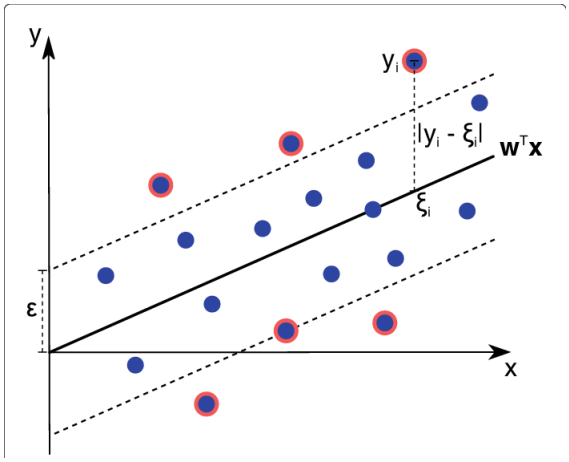

Figura 2: Visualização de *Support Vector Regression*

Árvores de Decisão

A árvore de decisão (DT - *Decision Tree*) é outro algoritmo mais utilizado em problemas de classificação. Porém, também é possível utilizá-lo em tarefas de regressão (GÉRON, 2017). O algoritmo funciona criando uma estrutura em árvore com nós internos com testes que são feitos para prever a qual nó folha o recurso será atribuído, retornando, no caso desse trabalho, um valor contínuo. Na Fig. 3 é mostrado como que esse algoritmo realiza sua regressão, ele divide os valores em setores e tenta fazer a previsão baseando-se nesses setores criados. Esses setores são criados a partir da configuração de hiperparâmetros da função, como por exemplo a profundidade máxima da árvore (*"max depth"*).

Floresta Aleatória

A floresta aleatória, do inglês *Random Forest* (RF), é um algoritmo semelhante às árvores de decisão, a diferença é que ao invés de computar apenas uma árvore de decisão, a RF calcula um grande número de árvores de decisão com profundidades diferentes para, em seguida, computar a média de todas as previsões para estimar o valor. Isso torna esse método um dos algoritmos supervisionados mais poderosos. Pode-se representar as árvores de decisão n unidas conforme descrito na Eq. 4, onde cada $f_n(x)$ representa um estimador (uma árvore) da floresta como na Fig. 4.

$$g(x) = f_0(x) + f_1(x) + f_2(x) + f_3(x) + ... + f_n(x) \tag{4}$$

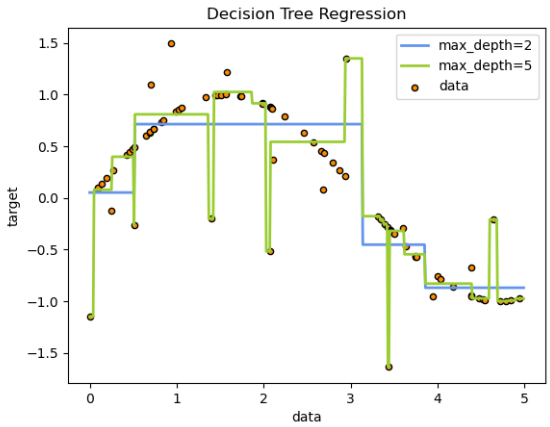

Figura 3: Visualização de *Decision Trees*

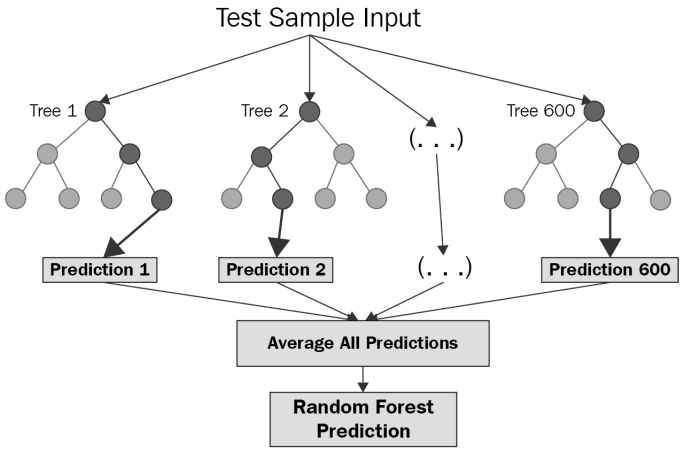

Figura 4: Visualização de Floresta Aleatória

Material e Métodos

O desenvolvimento das atividades neste projeto iniciou com o estudo dos conceitos básicos sobre tecnologia CMOS e FinFET, sobre o projeto de circuitos integrados e sobre simulação elétrica. Estes conhecimentos são fundamentais para o entendimentos dos passos de caracterização elétrica, de falhas de radiação e de influência de variabilidade nas métricas de atraso e potência.

A seguir, este relatório detalha os materiais e métodos adotados para inicialmente formular o fluxo para caracterização de uma célula NAND2, adotando técnicas de aprendizagem de máquina. Na Fig. 5 apresenta-se um diagrama que resume o processo de treinamento realizado. Assim, divide-se o processo em sete etapas principais iniciando com as simulações elétricas para extração das variáveis, seguindo pelo tratamento destas variáveis, realizando a normalização e também a separação destes dados em conjuntos de treinamento, validação e teste para os algoritmos avaliados. A Etapa quatro contempla o desenvolvimento e ajuste dos algoritmos considerados. A Etapa cinco realiza a análise dos algoritmos. Finalmente, é realizada a compilação dos resultados e também a disponibilização em uma interface online do modelo treinado. Os passos envolvidos neste fluxo serão detalhados nas próximas seções.

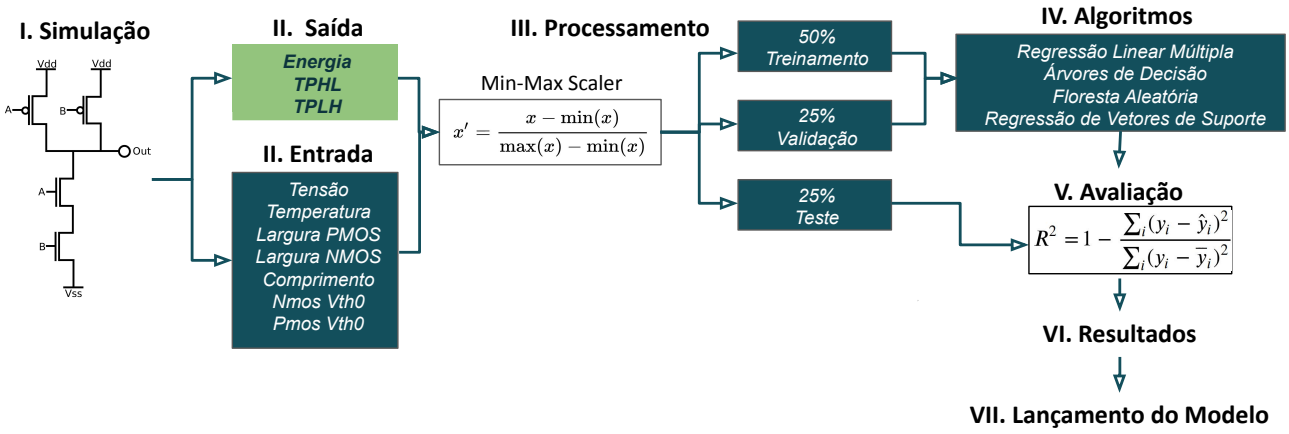

Figura 5: Diagrama do fluxo de treinamento

Extração de Variáveis

A caracterização elétrica de um circuito não-e prevê a obtenção de dados de energia e dos atrasos. Os tempos de propagação de subida e descida são utilizados para definir os atrasos dos circuitos. Estas variáveis sofrem a influência de diversos parâmetros da configuração do circuito tais como o tamanho dos dispositivos, a largura de canal, a capacitância de carga, a temperatura, a tensão de operação, e os efeitos de variabilidade impactam esses dados também dependendo dos parâmetros adotados. Para prever o tp$_HL$ (atraso de descida), o tp$_LH$ (atraso de subida) e a energia do circuito NAND2 CMOS, o primeiro passo foi definir claramente quais são as variáveis de entrada e seus valores. A Tabela 1 resume essas informações. Foram simulados os valores de nove variáveis de entrada: a tensão de limiar para dispositivos NMOS e PMOS pelo parâmetro Vth0 no modelo do dispositivo, a temperatura, a capacitância, a tensão de operação, a largura dos transistores (W) para dispositivos PMOS e NMOS e o comprimento do canal (L) para dispositivos PMOS e NMOS.

Tabela 1: Variáveis de entrada e de saída

Saída	Tp$_{HL}$	Tp$_{LH}$	Energia
Entrada	NMOS Vth0 PMOS Vth0 temperatura tensão W PMOS W NMOS L capacitância	NMOS Vth0 PMOS Vth0 temperatura tensão W PMOS W NMOS L capacitância	NMOS Vth0 PMOS Vth0 temperatura tensão W PMOS W NMOS L capacitância

Neste experimento, adotou-se o modelo de transistor para a tecnologia CMOS Bulk de alto desempenho de 16 nm fornecido pelo PTM (Wei Zhao; Yu Cao, 2006). O parâmetro Vth0 é variado para simular o impacto da variabilidade do processo devido à variabilidade geométrica e variabilidade aleatória de dopante na tensão de limiar do transistor. A variabilidade de processo é simulada utilizando o método de Monte Carlo, onde a alteração na tensão de limiar segue uma distribuição Gaussiana com 3σ e 10% de desvio padrão (ALIOTO; CONSOLI; PALUMBO, 2015). Todas as simulações elétricas adotaram o *software* de simulação Hspice da Synopsys. Predefiniu-se os valores de largura, comprimento, temperatura, capacitância e tensão, seguindo estas variações:

- **Temperatura (^oC)**: -25, 0, 25 50, 75 e 100
- **Largura PMOS (nm)**: 70, 140, 210
- **Largura NMOS (nm)**: 70 e 140, 210, 280, 340, 420
- **Comprimento do PMOS e NMOS (nm)**: 32 e 40
- **Tensão (V)**: 0.6, 0.7, 0.8 e 0.9
- **Capacitância (F)**: 1, 4, 8 e 16

Os valores da energia e dos tempos de propagação (Tp$_H L$ e Tp$_L H$) foram todos obtidos pelas simulações elétricas com base em todas essas variáveis. No final, foram feitas 80 simulações transientes de 20 ns com um passo de 0,1 ns, obtendo-se ao final um total de 276.480 observações. No entanto, devido à remoção de valores discrepantes que confundiriam os resultados dos modelos, o número final de observações ficou em 271.863.

Normalização das variáveis

Os resultados de energia e de atrasos para a tecnologia de 16 nm são na ordem de picosegundos e fentoJoules. Por isso, foi preciso fazer um redimensionamento em cada variável, já que, além de serem muito pequenas, alguns algoritmos (como o SVR) são sensíveis a essas escalas devido à forma como seu cálculo é realizado. Para esse redimensionamento, usou-se o método conhecido como normalização, que dimensiona os dados para o intervalo [0,1]. Matematicamente, é adotada a Eq. 5, na qual se subtrai do valor X o valor mínimo e divide-se esse resultado pela diferença entre o máximo e o mínimo.

$$X' = \frac{X - min(X)}{max(X) - min(X)} \tag{5}$$

Dessa forma, as distribuições das variáveis de saída ficaram como na Fig. 6, essas distribuições serão usadas para visualizar de forma mais intuitiva a performance de cada algoritmo.

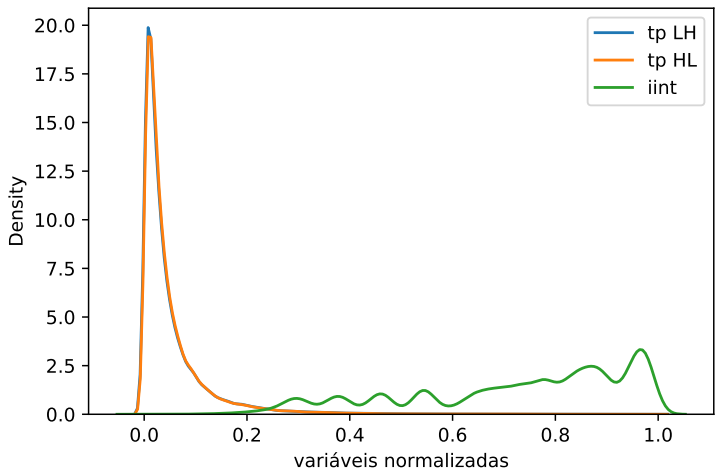

Figura 6: Distribuições das variáveis de saída normalizadas

Treinamento dos Algoritmos

A abordagem para prever as três variáveis de saída (tp$_H L$, tp$_L H$, energia) foi criar todos os quatro modelos para cada uma delas (Regressão Linear Múltipla, Regressão de Vetores de Suporte, Árvores de Decisão e Floresta Aleatória). Assim, ao final, foram treinados 12 modelos.

Criaram-se três cópias do conjunto de dados e, em seguida, o módulo de *model selection* do *sci-kit learn* foi utilizado para dividir o conjunto de dados entre o grupo de treinamento (50%), de validação (25%) e o de teste(25%).

As ferramentas utilizadas para manipular, visualizar, limpar, treinar e testar os dados foram o Colab do Google e as bibliotecas da linguagem python sklearn, pandas, NumPy, matplotlib e seaborn. Todos os algoritmos foram treinados utilizando a técnica conhecida como *cross validation* (validação cruzada). Ela é realizada para que o algoritmo faça previsões mais gerais e não tenha um *overfit* ou enviesamento nos dados que foram alimentados durante seu treinamento. Para isso, o *cross-validation* utiliza do conjunto de treinamento e do conjunto de validação para primeiro fazer avaliação de como o modelo está se comportando. Além disso, nesse processo criam-se diversas cópias do modelo que está sendo treinado e são variados seus hiperparâmetros para averiguar qual é o melhor modelo com a melhor configuração desses hiperparâmetros. Nos modelos trabalhados, apenas a Regressão Linear Múltipla não possui hiperparâmetros, logo os seguintes valores foram configurados para o *cross-valdation*:

- **Árvore de Decisão (DT) - *Max Depth***: $1, 5, 10, 25, 50$
- **Floresta Aleatória (RF) - *Max Depth***: $1, 5, 10, 25, 50$
- **Floresta Aleatória (RF) - *N Estimators***: $5, 25, 50, 100, 150$

Tanto para a Árvore de Decisão quanto para a Floresta Aleatória, *max depth* é a profundidade máxima que cada árvore terá. Os valores de *N Estimators* da Floresta Aleatória são quantos estimadores, ou seja, quantas árvores serão criadas para o modelo. Já os valores *Gamma* e os valores *C* para a Regressão de Vetores de Suporte são referentes ao valor γ da Eq.2 e à regularização do modelo, respectivamente. Ao final do cross validation, foram obtidos os valores da Tabela 2. Não foram colocados os valores do algoritmo SVR pois o treinamento utilizando essa metodologia provou-se muito extensa (mais de três dias) e com os recursos pessoais e online utilizados pelo bolsista não foi possível gerar esses dados atualmente, portanto optou-se por seguir com o valor padrão de $C = 1$ $Gamma = \frac{1}{\sigma^2}$ para todas as variáveis de saída.

Tabela 2: Resultados do cross-validation

	Hiperparâmetros		
Variável de Saída	DT - *Max Depth*	RF - *Max Depth*	RF - *N Estimators*
tpHL	5	5	100
tpLH	5	5	150
Energia	5	5	150

Medida de Performance

Para avaliar a adequação de cada modelo, precisa-se avaliar o erro entre o valor previsto e o valor desejado. Neste trabalho, usou-se a Raiz do Erro Quadrático Médio (*Root Mean Square Error* - RMSE) e o coefieciente de determinação (conhecido como "r-quadrado"ou "r-dois- R^2) como medidas de desempenho, uma vez que o RMSE é o mais comumente usado

em problemas de regressão e o R^2 é mais facilmente interpretado. O RMSE dá pesos maiores para erros grandes, o que significa que é mais sensível a outliers (GÉRON, 2017), por isso foi importante fazer a remoção desses anteriormente ao treinamento. Ele é calculado usando a norma euclidiana, conforme apresentado na Eq. 6, onde m é o número de instâncias no conjunto de dados, $x^{(i)}$ e $y^{(i)}$ são vetores de todos valores de características e o valor desejado da característica i^{th} do conjunto de dados, **X** é a matriz dos vetores de características e h é a função modelo para a previsão (GÉRON, 2017).

$$RMSE(\mathbf{X}, h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2} \tag{6}$$

Já o R^2 é a variância explicada (Eq. 7), ou seja, ele diz o quão bem o modelo criado está conseguindo explicar a variação total no valor alvo e fica no intervalo entre 0 e 1. Um valor R^2 de 1 significa que o modelo foi capaz de explicar 100% do comportamento modelado, então valores baixos de R^2 significam que o modelo está mal ajustado e valores altos significam um bom ajuste do modelo (KANE, 2017)

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y}_i)^2} \tag{7}$$

Disponibilização do Modelo

O melhor algoritmo foi disponibilizado na internet (Fig. 8) no sítio eletrônico: https://electrical-characterization.herokuapp.com (acessível também pelo QR code da Fig. 7). O site foi construído a partir da biblioteca Streamlit da linguagem python e nele é possível simular os valores desejados para que o algoritmo faça suas predições em tempo real.

Figura 7: QR code para o site

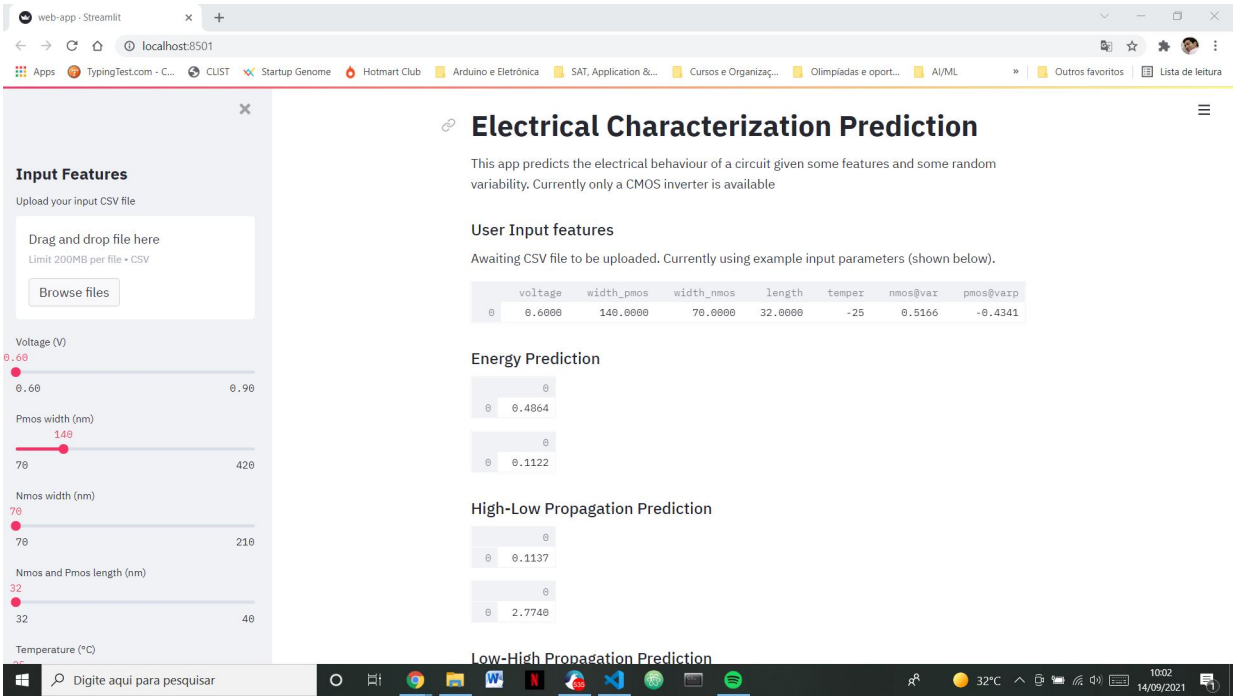

Figura 8: Site com o modelo lançado

Resultados

Depois do treino e validação dos modelos em 75% dos dados, verificou-se os desempenhos no conjunto de teste a partir da função de custo RMSE e do R^2 conforme descrito na Tabela 3 para cada variável, onde tpHL é o atraso de descida, tpLH é o atraso de subida e a terceira coluna apresenta os resultados de energia.

Tabela 3: Resultados do RSME para tpHL, tpHL e Energia (NAND2 CMOS)

Modelo	**Variáveis alvo**					
	tpHL		**tpLH**		**Energia**	
	RMSE	R²	RMSE	R²	RMSE	R²
RF	0.0012	0.9995	0.0009	0.9998	0.0022	0.9998
MLR	0.0395	0.5363	0.0483	0.5892	0.0354	0.9730
SVR	0.0481	0.3126	0.0431	0.6730	0.0351	0.9734
DT	0.0016	0.9991	0.0012	0.9997	0.0033	0.9997

Predição do Atraso de Propagação de Descida (TP$_HL$)

Como pode-se ver na Tabela 3, para o Tp$_HL$, os maiores valores do R^2 são 99.95% e 99.91%, que são os valores para os modelo do Floresta Aleatória (RF) e do Árvores de Decisão (DT). Este é um ótimo resultado, fato que também pode ser constatado pela Fig. 9, na qual mal pode-se ver a diferença entre a curva predita e a curva esperada. No entanto, quando os resultados dos outros algoritmos são analisados, os valores são diferentes. Os R^2 para a Regressão Linear Múltipla (MLR) e a Regressão de Vetores de Suporte (SVR) são os menores

da tabela para o Tp$_HL$. Na Fig. 9 pode-se observar as distribuições das previsões de cada modelo comparadas com a distribuição correta.

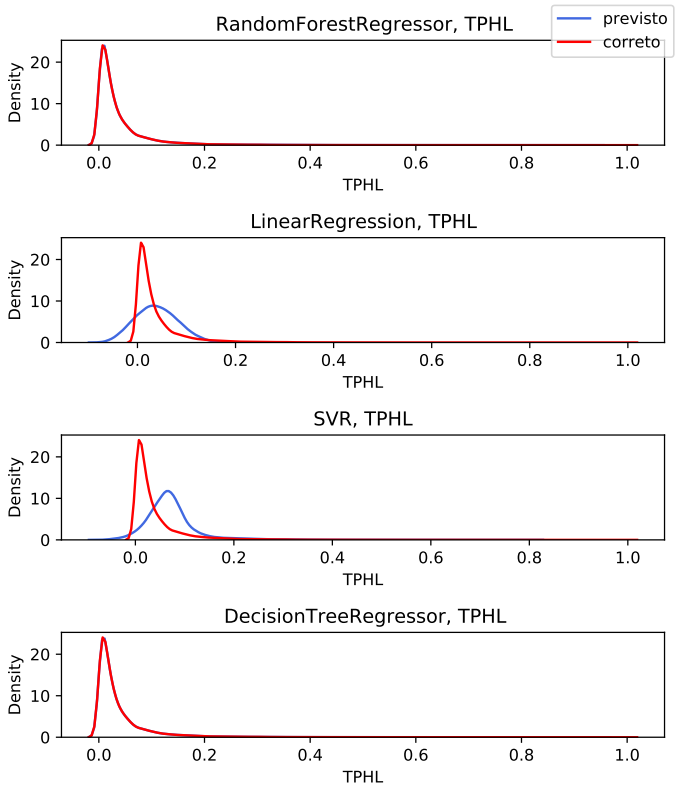

Figura 9: TPHL

Predição do Atraso de Propagação de Subida (TP$_LH$)

Os resultados para o Tp$_LH$ são muito semelhantes aos do Tp$_HL$. Os números mostram que o RF ainda é o melhor método para prever a variável desejada e o MLR o pior. Ademais, as semelhanças de comportamento dos valores de R^2 do DT e do RF permanecem as mesmas da previsão do Tp$_HL$. Na Fig. 10 é possível constatar esses valores de forma mais intuitiva com a curva de distribuição esperada e a curva predita por cada modelo.

Predição da Energia

Para a energia, o melhor algoritmo foi, novamente, o RF, seguido do DT, sendo o SVR o penúltimo e o MLR o último. O menor R^2 de toda tabela foi dado pelo MLR, sendo igual a 97.30%. Na Fig. 11, pode-se observar novamente o mesmo padrão de comportamento encontrado para as variáveis de atraso.

A pior performance do algoritmo MLR pode ser justificada pela forma como a correlação entre nossas variáveis é dada. Essa correlação pode ser observada na Fig. 12, na qual se apresentam os valores do coeficiente de correlação linear de Pearson. Quanto mais perto de 1 ou -1 é o valor de correlação, mais é possível explicar as variáveis por uma relação

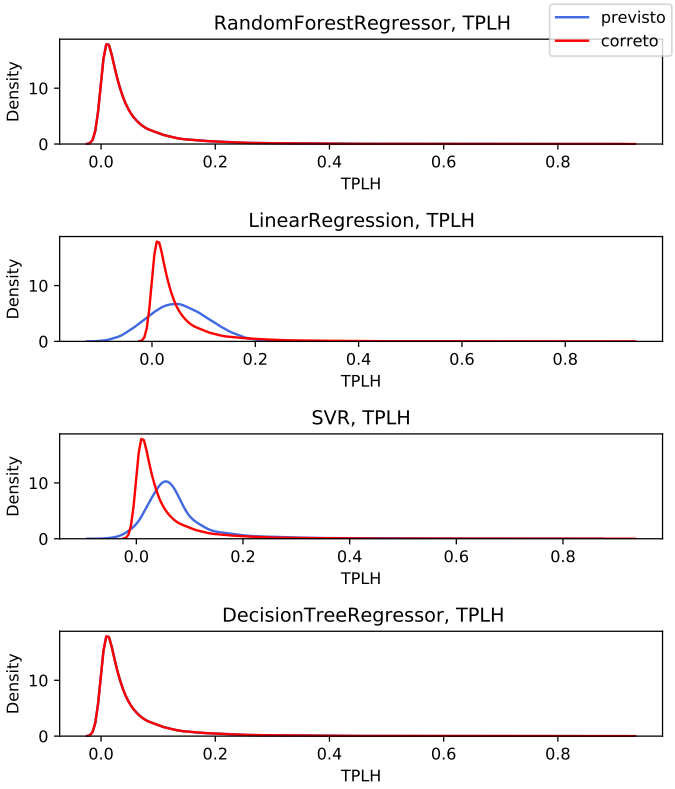

Figura 10: TPLH

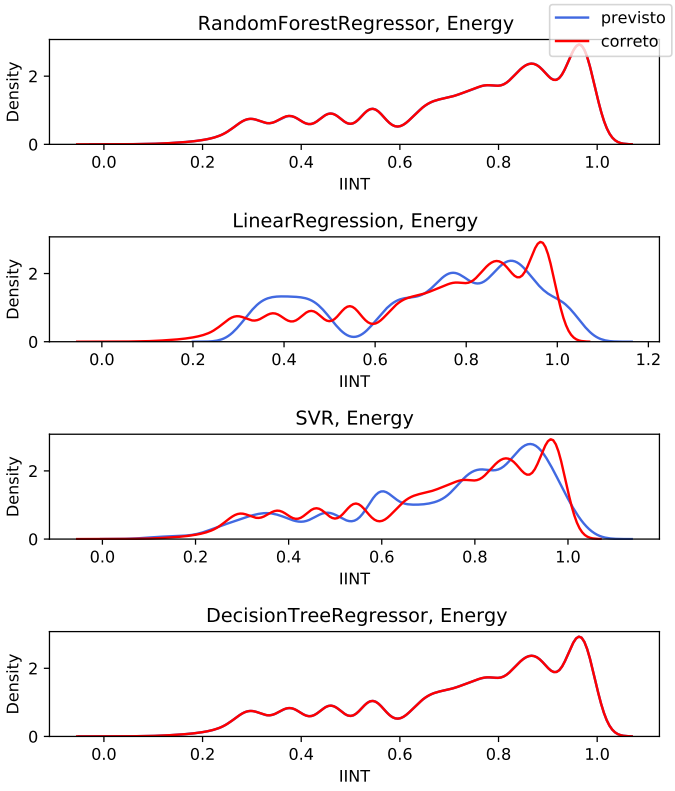

Figura 11: Energy

linear (positiva ou negativa). Portanto, pode-se ver que não há muito em que o algoritmo de Regressão Linear Múltipla possa se basear para ter uma melhor performance.

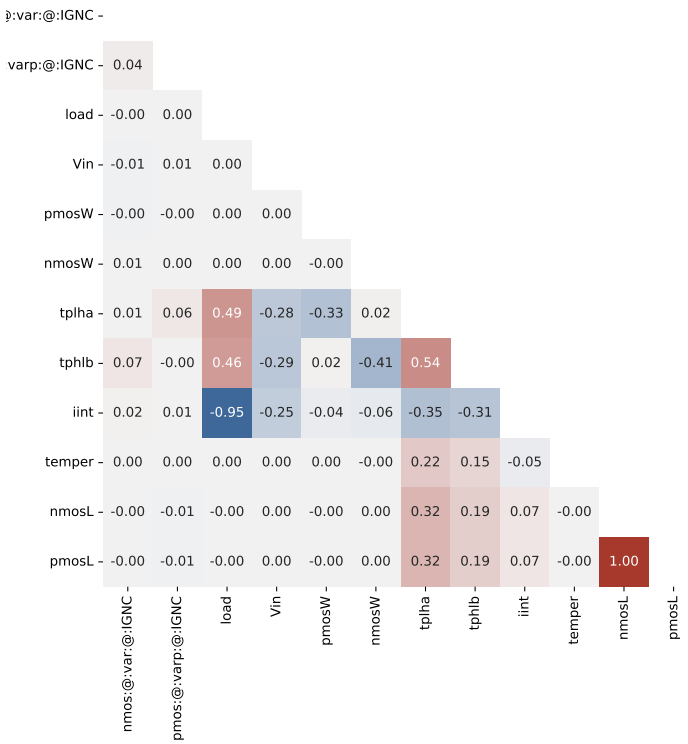

Figura 12: Coeficientes da Correlação Linear de Pearson

Tempo de Inferência

Uma análise final foi realizada para avaliar o *trade-off* entre a precisão dos modelos e tempo de execução por inferência de cada um. Pelos dados gerados, os modelos DT superaram os de RF, embora RF apresentem valores um pouco superiores de R^2. Como exemplo, uma única simulação elétrica para a porta NAND2 é executada por 17,5 s na ferramenta HSPICE, em comparação com 0,0006 s no modelo DT e 0,0398 s no modelo RF.

Conclusões

Neste trabalho, foram analisados quatro tipos de algoritmos de aprendizado de máquina para a tarefa de regressão de prever o comportamento elétrico de um circuito NAND2 CMOS. Treinaram-se os algoritmos com os dados da simulação de Monte Carlo usando o software Hspice.

Os resultados mostram que, para todas as três variáveis de saída (tp_{HL}, tp_{LH} e energia), o algoritmo de melhor desempenho foi o Floresta Aleatória, seguido do Árvores de Decisão, ou seja, os melhores algoritmos foram aqueles que baseiam suas técnicas em estruturas de árvores. Contudo, para o objetivo desse trabalho, considerando que o tempo que os algoritmos levam para fazer suas predições deve ser o menor o possível comparado com a simulção tradicional, o algortimo que equilibra predição correta com menor tempo de inferência foi o Árvore de Decisão.

Os resultados inferiores de performance do algoritmo de Regressão Linear Múltipla podem ser justificados salientando-se que muitas das variáveis não possuem uma correlação linear. Quanto ao SVR, trata-se também de um algoritmo muito poderoso e que, talvez com o uso de outros *kernels* como o polinomial, é possível chegar um erro menor. Contudo, mesmo que se chegue a essa melhor performance no SVR, esse possui um problema que não é possível evitar: o elevado tempo de treinamento, tanto com e sem a técnica de *cross-validation*. Como o conjunto de dados de simulação é grande e a tendência para trabalhos futuros é só aumentar, o SVR não escala tão bem para fazer um treinamento rápido, o que foge do objetivo do trabalho que é reduzir o tempo de simulação de circuitos por meio de algoritmos de *machine learning*.

Em trabalhos futuros pretende-se explorar outros modelos mais complexos e mais atuais da literatura como arquiteturas de redes neurais. Ademais, pretende-se simular outros circuitos mais complexos, como portas XOR e *Full-adder*.

Portanto, o aprendizado de máquina é um ferramenta interessante e poderosa para prever o comportamento de circuitos elétricos, mesmo em sua técnica mais básica, e reduzir o longo processo de simulação tradicional. Esse fluxo de treinamento pode ser usado futuramente por fabricantes que desejam disponibilizar seus dispositivos sem a necessidade de abrir mão dos direitos da sua propriedade intelectual. Ademais, pode-se a partir dos modelos treinados criar ferramentas educativas de fácil uso para alunos de fluxo de síntese de circuitos (como é o protótipo construído e disponibilizado no endereço eletrônico https://electrical-characterization.herokuapp.com/)

O trabalho neste projeto possibilitou ao bolsista a produção e a apresentação de um artigo em evento internacional e um pôster em um evento nacional:

- setembro 2021: *Electrical Behavior Prediction Of An Inverter Using Machine Learning Algorithms* no 11º IEEE CASS Rio Grande do Sul Workshop 2021 (CASSW-RS 2021);
- junho 2022: *Exploring Machine Learning Algorithms for Electrical Behavior Prediction: The CMOS Inverter Case Study* no Simpósio Internacional de Circuitos Integrados e Projeto de Sistemas (SBCCI 2022);

Avaliação do Aluno em Relação ao PIBIC

O bolsista iniciou seus trabalhos graças ao convite da ilustre professora Cristina Meinhardt, que sempre apoiou o projeto com dicas, direcionamentos e tutoriais, além de ser uma ótima orientadora quando há imprevistos e sobrecarregamento de tarefas. Por meio do Programa Institucional de Bolsas de Iniciação Científica (PIBIC), não apenas foi possível que o bolsista tivesse mais contato com a dinâmica de um projeto de pesquisa em sua totalidade (desde o planejamento até a discussão com avaliadores de conferências), mas também criou-se a oportunidade de conhecer novos professores e contatos que ajudarão futuramente com a continuação do trabalho. Graças ao PIBIC, o bolsista pode ter a certeza de que o meio acadêmico brasileiro, por muitas vezes negligenciado por terceiros, é vasto, diverso e muito rico. Rico em talentos, rico em projetos e rico em potencial. O bolsista possui uma enorme gratidão à professora Cristina Meinhardt e ao PIBIC por terem aberto essa oportunidade que continuará a dar bons frutos e que contribuirá não só para a formação de um pesquisador preparado para a pós-graduação mas também para a comunidade tecnocientífica como um todo.

UNIVERSIDADE FEDERAL DE SANTA CATARINA

Pró-Reitoria de Extensão

Certificado

Certificamos que Gabriel Lima Jacinto
CPF 074.618.471-98
atuou como Participante do(a) Minicurso - Consciência da ciência ou sociedade sem ciência?
realizado no período de 02/03/2022 a 08/03/2022
Carga horária: 8 horas, com aproveitamento e com frequência suficiente
Esta atividade está amparada pelo Projeto de Extensão intitulado: Consciência da ciência ou sociedade sem ciência?

TÓPICOS ABORDADOS
Módulo 1 - Conceitos fundamentais sobre o método científico
Histórico e modus operandi. Comunidade científica e consenso. Erros, fraudes, ética. A importância da diversidade na ciência. Por que confiar na ciência?

Módulo 2 - Pseudociências e desinformação
Pseudociências e fenômeno da pós-verdade. Negacionismo científico e fake news. Analfabetismo científico e relação cientistas-sociedade. O que pensa a sociedade sobre a ciência? Quem são os inimigos da ciência?

Módulo 3 - Divulgação Científica
Noções de popularização da ciência. A importância da divulgação científica. Manual de sobrevivência na divulgação científica. Como fazer divulgação científica

Coordenador: Maique Weber Biavatti
Protocolo: 202201212

Este certificado dispensa assinatura
UFSC - PROEX
Campus Reitor João David Ferreira Lima
Florianópolis - Santa Catarina - Brasil
CNPJ: 83.899.526/0001-82

A autenticidade do documento pode ser verificada no site: http://autenticidade.ufsc.br, informando a chave: CERT-5808-5348-1009-2694

Referências Bibliográficas

ALIOTO, M.; CONSOLI, E.; PALUMBO, G. Variations in nanometer cmos flip-flops: Part i—impact of process variations on timing. *IEEE Transactions on Circuits and Systems I: Regular Papers*, IEEE, v. 62, n. 8, p. 2035–2043, 2015.

Barboza, E. C.; Shukla, N.; Chen, Y.; Hu, J. Machine learning-based pre-routing timing prediction with reduced pessimism. In: *2019 56th ACM/IEEE Design Automation Conference (DAC)*. [S.l.: s.n.], 2019. p. 1–6.

Chatterjee, A.; Croley, D.; Ramamurti, V.; Kui-Yu Chang. Application of machine learning to manufacturing: results from metal etch. In: *Nineteenth IEEE/CPMT International Electronics Manufacturing Technology Symposium*. [S.l.: s.n.], 1996. p. 372–377.

GÉRON, A. *Hands-on machine learning with Scikit-Learn and TensorFlow concepts, tools, and techniques to build intelligentsystems*. O'Reilly Media, 2017. Paperback. Disponível em: <http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/1491962291>.

KAHNG, A. B. Reducing time and effort in ic implementation: A roadmap of challenges and solutions. In: *Proceedings of the 55th Annual Design Automation Conference*. New York, NY, USA: Association for Computing Machinery, 2018. (DAC 18). ISBN 9781450357005. Disponível em: <https://doi.org/10.1145/3195970.3199854>.

Kahng, A. B.; Mallappa, U.; Saul, L. Using machine learning to predict path-based slack from graph-based timing analysis. In: *2018 IEEE 36th International Conference on Computer Design (ICCD)*. [S.l.: s.n.], 2018. p. 603–612.

Kaintura, A.; Foss, K.; Couckuyt, I.; Dhaene, T.; Zografos, O.; Vaysset, A.; Sorée, B. Machine learning for fast characterization of magnetic logic devices. In: *2018 IEEE Electrical Design of Advanced Packaging and Systems Symposium (EDAPS)*. [S.l.: s.n.], 2018. p. 1–3.

KANE, F. *Hands-on data science and python machine learning*. 1. ed. Birmingham, UK: Packt Publishing, 2017.

Lee, J. K.; Ko, K.; Shin, H. Analysis on process variation effect of 3d nand flash memory cell through machine learning model. In: *2020 4th IEEE Electron Devices Technology Manufacturing Conference (EDTM)*. [S.l.: s.n.], 2020. p. 1–4.

Wei Zhao; Yu Cao. New generation of predictive technology model for sub-45nm design exploration. In: *7th International Symposium on Quality Electronic Design (ISQED'06)*. [S.l.: s.n.], 2006. p. 6 pp.–590.

Florianópolis, 31 de outubro de 2022.