Universidade Federal de Pernambuco
Centro de Informática

Graduação em Engenharia da Computação

Aplicação de Aprendizado de Máquina para Previsão de Fluxo de Caixa em ATMs

Gedson Santos de Melo

Recife, 2018

Universidade Federal de Pernambuco
Centro de Informática

Gedson Santos de Melo

Aplicação de Aprendizagem de Máquina para Previsão de Fluxo de Caixa em ATMs

Monografia apresentada como requisito parcial
para a obtenção do grau de
Bacharel em Engenharia da Computação

Orientador: Ricardo Bastos C. Prudencio

Recife, 2018

Abstract

This work is focused on the problem that financial institutions have in predicting how much money the ATMs need to be supplied. This problem affects the said institutions because if the ATM is overloaded, this money will not be yielding, and also affects customers because if stocked with little money users will not be able to withdraw. With the use of machine learning it is possible to realize a forecast and get the value that the ATM will need. As algorithms, we used supervised learning in the Multilayer Perceptron (MLP) and Long Short-Term Memory (LSTM) neural networks. For the analysis the Mean absolute percentage error (MAPE), Wilcoxon test and Moving Average were used. The MAPE for MLP resuts ranged from 18% to 34%, for LSTM it ranged from 17% to 36% and for Moving Average it ranged from 43% to 60%. This observed variation was resulted from applying the algorithms on data from different ATMs and the change in the number of steps of supervised learning. By compiling the analyzes, it is seen that MAPE obtained values similar to those of the reviewed articles and the Wincoxon test showed that the difference of the values of the predictions with the expected values are not statistically significant.

Resumo

Esse trabalho é voltado para o problema que as instituições financeiras têm em prever qual quantidade de dinheiro os *Automatic Teller Machine* (ATM) precisam para serem abastecidos. Esse problema afeta tanto as instituições, pois se o ATM for sobrecarregado esse dinheiro não estará rendendo, quanto os clientes, pois se for abastecido com pouco dinheiro os usuários não vão poder sacar. Com a utilização de aprendizagem de máquina é possível realizar uma previsão e obter o valor que o ATM irá precisar. Como algoritmos, foram usados aprendizagem supervisionada nas redes neurais *Multilayer Perceptron* (MLP) e *Long Short-Term Memory* (LSTM). Para a análise dos resultados foram usados o *Mean absolute percentage error* (MAPE), o teste de *Wilcoxon* e a Média Móvel. O MAPE para o MLP variou de 18% a 34%, para o LSTM variou de 17% a 36% e para a Média Móvel variou de 43% a 60%. Esta variação observada é resultado da aplicação dos algoritmos em dados de ATMs diferentes e a mudança nos números de passos da aprendizagem supervisionada. Compilando as análises é visto que o MAPE obteve valores parecidos com os dos artigos revisados e o teste de *Wilcoxon* mostrou que a diferença dos valores das previsões com os valores esperados não é estatisticamente significante.

Conteúdo

Lista de Figuras

Lista de Tabelas

1 Introdução

1.1 Motivação

Os ATMs (Automatic Teller Machine) ou caixas eletrônicos são dispositivos gerenciados por instituições que disponibilizam aos clientes um método simples para conduzir transações financeiras em um espaço público com quase nenhuma intervenção humana[1]. Segundo a ATMIA (ATM Industry Association), em 2015 já existiam 3,2 milhões de ATM instalados em todo o mundo e que esse número chegará em 4 milhões em 2021[2].

Alguns bancos normalmente mantêm até 40% mais dinheiro em seus caixas eletrônicos do que o necessário, embora muitos especialistas consideram que o excesso de caixa de 15% a 20% seja suficiente[3]. Como os bancos ganham dinheiro usando o que é depositado[4], ao fazer empréstimo, por exemplo, esse dinheiro estando parado em ATMs deixa de gerar lucro para o banco.

É difícil para o banco prever quanto e quando os ATMs precisam ser reabastecidos, sendo esse trabalho feito manualmente por empresas de abastecimento. Uma das formas de fazer essa previsão de fluxo de caixa é usar aprendizado de máquina (machine learning). Aprendizado de máquina, em inteligência artificial, é uma disciplina relacionada com a implementação de software que pode aprender de forma autônoma.

Sistemas especialistas e programas de mineração de dados são os aplicativos mais comuns a usar algoritmos de aprendizado de máquina para melhorar seu desempenho. Entre as abordagens mais comuns estão o uso de redes neurais artificiais e algoritmos genéticos[5].

1.2 Objetivo

A pretensão desse Trabalho de Graduação é usar aprendizado de máquina com as redes neurais Long Short-Term Memory e Multilayer Perceptron para prever o fluxo de caixa em ATM, usando um *dataset* coletado de um banco da Índia (City Union Bank Limited).

A previsão provê ao banco a funcionalidade de indicar a empresa de abastecimento exatamente quais caixas devem ser reabastecidos e qual o valor que cada um necessita, com isso tanto o banco ganha, não deixando dinheiro parado, quanto os clientes ganham, pois sempre terão dinheiro disponível para realização de seus saques.

1.3 Estrutura do Trabalho

Este trabalho é organizado da seguinte maneira. O Capítulo 2 refere-se à revisão de literatura e é responsável por apresentar todos os conceitos básicos. Também fornece a base necessária para entender a teoria associada às áreas de estudos relacionadas a este trabalho e a motivação por trás de cada uma delas. O Capítulo 3 explica em detalhes o método proposto. Ele decifra os componentes, etapas e suposições feitas durante a implementação e descreve os conjuntos de dados utilizados neste trabalho. O estudo experimental pode ser visto no Capítulo 4, bem como a forma como os experimentos foram feitos, aplicação das métricas de avaliação escolhidas e discute os resultados obtidos. O Capítulo 5 resume o que pode ser aprendido com este trabalho, se os objetivos foram alcançados e sugere algumas melhorias a serem consideradas em trabalhos futuros.

2 Revisão da Literatura

2.1 Estado da Arte

Esta seção visa discutir alguns trabalhos com os mesmos objetivos deste trabalho, o primeiro artigo proposto por Rimvydas Simutis, Darius Dilijonas e Lidija Bastina[6], onde dois diferentes métodos são usados para fazer a previsão. O primeiro método é baseado na rede neural artificial flexível (ANN) e teve um erro percentual absoluto médio (MAPE) de 15-28% na previsão. O segundo método de previsão emprega a máquina de vetores de suporte (SVM) e teve um MAPE de 17-40%. Os erros citados foram obtidos quando os algoritmos foram submetidos a dados reais de ATM.

O próximo trabalho foi feito por Nidhi Arora e Jatinder Kumar[1]. Ele sugere uma aplicação de uma rede *fuzzy ARTMAP*[7] para analisar e prever o fluxo de caixa em ATM, o modelo teve um MAPE de 4-14% em variação do erro que vem da mudança do tamanho do *dataset* e de parâmetros usados no modelo.

Por último, no trabalho publicado por Cristián Ramírez e Gonzalo Acuña[8] foram usados os algoritmos Multilayer Perceptron (MLP) e Vetor de Suporte de Mínimos Quadrados (LS-SVM) com dois tipos de entradas diferentes que são as não-lineares auto-regressivas exógenas (NARX) e média móvel não-linear com entradas exógenas (NARMAX). No artigo foi usado o erro percentual absoluto médio simétrico (SMAPE), os resultados foram 19,80% para a MLP usando NARX, 20,55% para MLP usando NARMAX e 24,35% para LS-SVM usando NARX.

2.2 Aprendizagem de Máquina

O aprendizado de máquina geralmente se refere a alterações em sistemas para executar tarefas associadas à Inteligência Artificial (IA). Tais tarefas envolvem reconhecimento, diagnóstico, planejamento, previsão, classificação e etc. As mudanças podem ser aprimoramentos para sistemas já em execução ou base para novos sistemas[9]. Uma definição mais precisa e formal dada por Mitchell[10]:

> Um programa aprende a partir da experiência *E* relacionada a alguma classe de tarefas *T* e medidas de performance *P*. E usa sua performance em *T* para melhorar a experiência *E.*

Há muitas razões para afirmar que aprendizado de máquina é importante. Pode-se citar que a realização de aprendizado em máquinas ajuda no entendimento de como os animais e humanos aprendem. Mas há razões importantes na engenharia também. Como é listado por Nilsson[9]:

- Algumas tarefas não podem ser bem definidas exceto através de exemplos. Sistemas capazes de aprender as relações contidas nos exemplos e generalizar para outras instâncias do problema são sistemas que têm desempenho melhor;

- Com uma grande quantidade de dados podem haver relações desconhecidas que se deseja explicitar. Sistemas com aprendizagem são capazes de revelar tais relações;

- Algumas características do ambiente em que o sistema será utilizado podem ser desconhecidas durante o projeto e implementação do sistema. Um sistema capaz de adaptar-se ao ambiente tem potencialmente maior capacidade de obter melhor eficiência;

- A quantidade de conhecimento disponível para determinada tarefa pode ser excessivamente grande para serem codificadas explicitamente. Sistemas que aprendam estes conhecimentos de forma automática tornam-se necessários;

- Ambientes se alteram com o tempo. Sistemas que se adaptam às mudanças exigem menos esforço de manutenção;

Há inúmeros algoritmos de aprendizado de máquina e eles são divididos em três categorias: Aprendizagem Supervisionada, Aprendizagem Não Supervisionada e Aprendizagem por Reforço. Os principais algoritmos voltados à aprendizagem supervisionada, que é o foco deste trabalho, são: Regressão Linear, Árvore de Decisão, Máquina de Vetor de Suporte (SVM), Rede neural, Naive Bayes, Algoritmo k-Nearest Neighbors (kNN) e Floresta Aleatória. E alguns desses podem se subdividir como as redes neurais. Neste trabalho vamos tratar de duas dessas subdivisões: Long short-term memory (LSTM) e Multilayer perceptron (MLP).

2.3 Multilayer Perceptron (MLP)

A unidade básica de computação em uma rede neural é o neurônio, frequentemente chamado de nó ou unidade. Recebe entrada de uma fonte externa e calcula uma saída. Cada entrada tem um peso associado (*w*), que é atribuído com base em sua importância relativa para outras entradas[11].

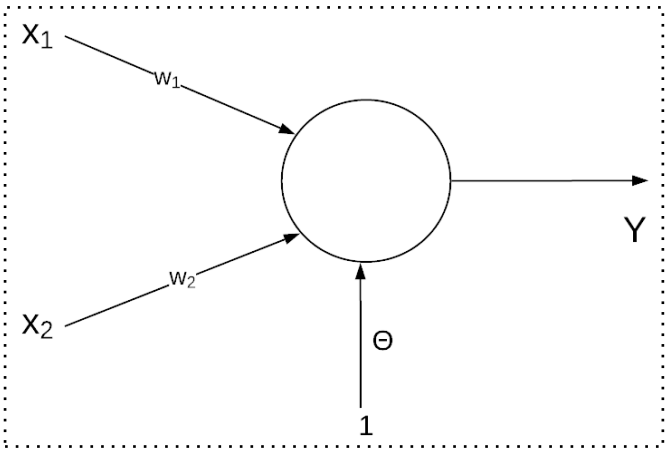

Figura 2.1: Modelo de perceptron simples

A rede acima aceita entradas numéricas X_1 e X_2 e possui pesos w_1 e w_2 associados a essas entradas. Além disso, há outra entrada *1* com peso Θ (chamado *Bias*) associado a ela, como é mostrado na Figura 2.1.

Porém, este modelo é incapaz de resolver problemas não linearmente separáveis, o que reduz em muito a aplicabilidade deste algoritmo[12]. Com isso foi necessária uma solução para este problema no que resultou no desenvolvimento do algoritmo de treinamento backpropagation[13].

Uma MLP backpropagation contém uma ou mais camadas ocultas, além de uma entrada e uma camada de saída. A Figura 2.2 mostra um perceptron multicamada com uma única camada oculta. Observe que todas as conexões possuem pesos associados a elas, mas apenas três pesos (w_0, w_1, w_2) são mostrados na figura.

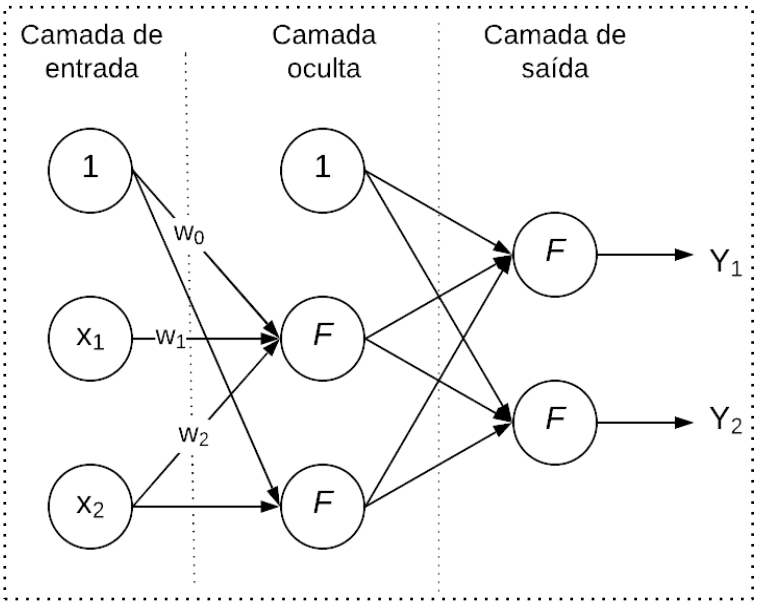

Figura 2.2: Modelo de um Multilayer perceptron com uma camada oculta

Camada de entrada: A camada de entrada da Figura 2.2 possui três nós. O nó *Bias* tem um valor de *1*. Os outros dois nós tomam X_1 e X_2 como entradas externas que são valores numéricos, dependendo do conjunto de dados de entrada. Nenhum cálculo é realizado na camada de entrada, portanto, as saídas dos nós na camada de entrada são *1*, X_1 e X_2, respectivamente, que são inseridas na camada oculta.

Camada oculta: A camada oculta da Figura 2.2 também possui três nós com o nó *Bias* tendo uma saída de *1*. A saída dos outros dois nós na camada oculta depende das saídas da camada de entrada (*1,* X_1, X_2), bem como da saída dos pesos associados às conexões (arestas). A Figura 2.2 mostra o cálculo da saída de um dos nós ocultos. Da mesma forma, a saída de outro nó oculto pode ser calculada. Na equação (1) é definido F apresentado na Figura 2.2 com *f* referindo-se à função de ativação. Essas saídas são então alimentadas para os nós na camada de saída.

$$F = f(w_0 * 1 + w_1 * x_1 + w_2 * x_2), \qquad (1)$$

Camada de saída: A camada de saída da Figura 2.2 possui dois nós que recebem entradas da camada oculta e executam cálculos semelhantes. Os valores calculados (Y_1 e Y_2) como resultado desses cálculos atuam como saídas da MLP.

Dado um conjunto de características $X = (x_1, x_2, \ldots)$ e um alvo Y, uma MLP pode aprender a relação entre os recursos e o alvo, tanto para classificação como para regressão.

2.4 Long short-term memory (LSTM)

Para entender a LSTM é necessário um conhecimento básico sobre redes neurais recorrentes (RNN). Nas RNN o estado anterior da rede influencia a saída, de modo que a rede também tem uma "noção de tempo". Consegue-se esse efeito com um *loop* na saída da camada para sua entrada, como podemos ver na Figura 2.3.

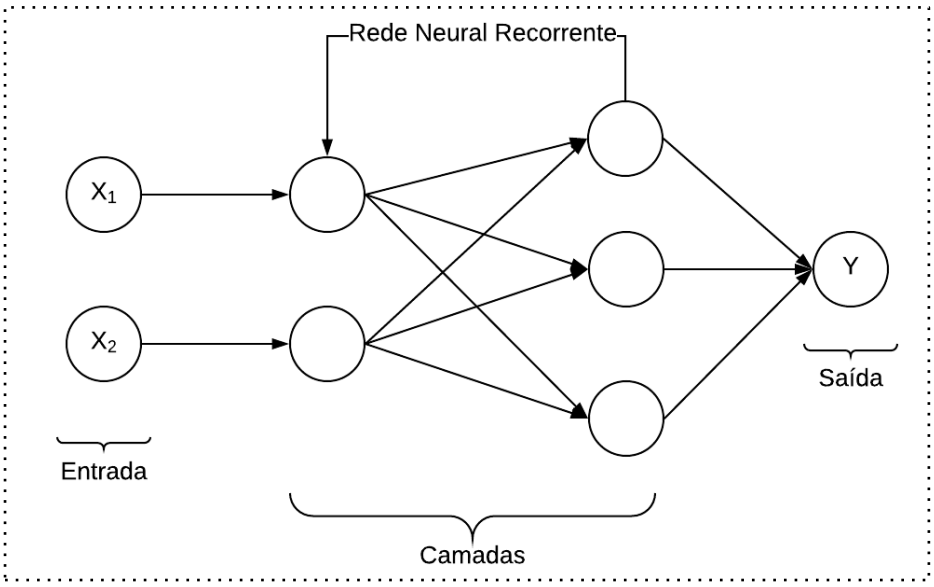

Figura 2.3: Representação de uma RNN

Isso dá a ideia de que as RNN podem conectar informações anteriores à tarefa atual, com o uso das informações anteriores, que podem ajudar o entendimento das informações atuais. Note na Figura 2.4 que cada célula *A* recebe uma entrada *X* e a saída da célula anterior.

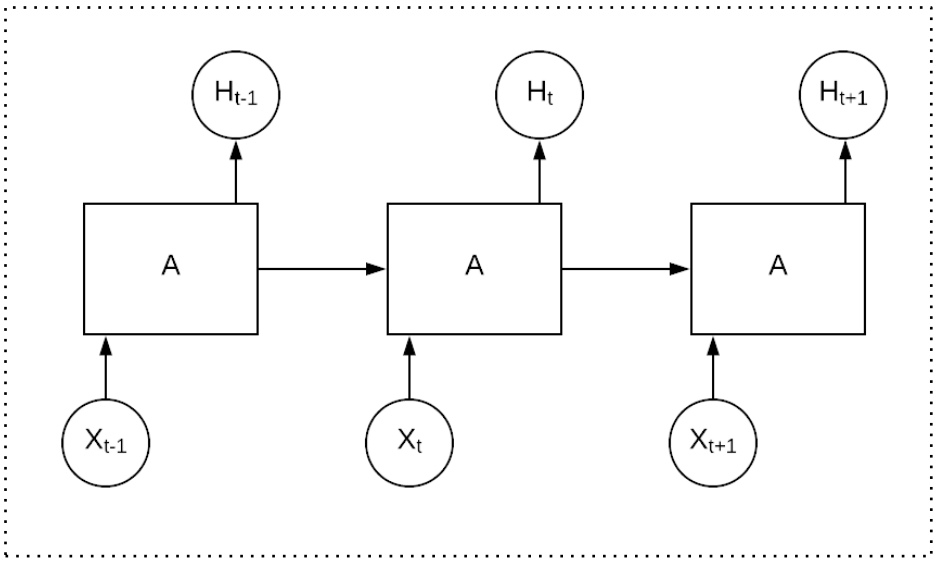

Figura 2.4: Representação das células de uma RNN

Por exemplo, um modelo que tem como entrada palavras e tenta prever a próxima. Se o modelo tenta prever a última palavra da frase: *O sofá está na sala*, não precisamos de mais nenhum contexto pois é óbvio que a próxima palavra será *sala*. Nesses casos, a distância entre a informação importante e a palavra que se tenta prever é pequena. Mas há casos que precisamos de mais contexto, por exemplo, usando o mesmo modelo para prever a frase: *Eu nasci no Brasil, por isso, minha língua nativa é o português*. Perceba que a frase não é só grande, mas a informação relevante para a previsão da palavra *Português* está consideravelmente distante[14].

Long short-term memory ou rede de memória de longo prazo é um tipo especial de RNN, capaz de guardar informações de longo prazo. Ele foi introduzido por Hochreiter & Schmidhuber[15].

A LSTM fará o mesmo que uma RNN, mas com uma memória maior. Como pode ser visto na Figura 2.5 cada célula *A* da LSTM recebe mais de uma informação da célula anterior, a saída e uma “memória” da célula.

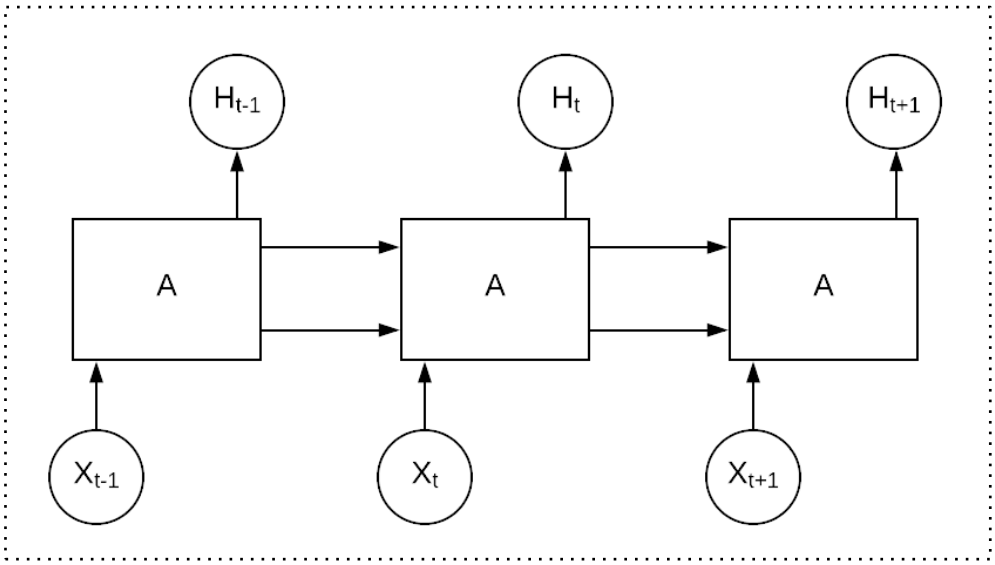

Figura 2.5: Representação das células de uma LSTM

Na Figura 2.6 vemos uma célula (A) da LSTM em detalhes que contém os seguintes componentes:

1. Porta de esquecimento *f*
2. Porta candidata *C’*
3. Entrada *I*
4. Saída *O*
5. Estado *H*
6. Estado de memória *C*

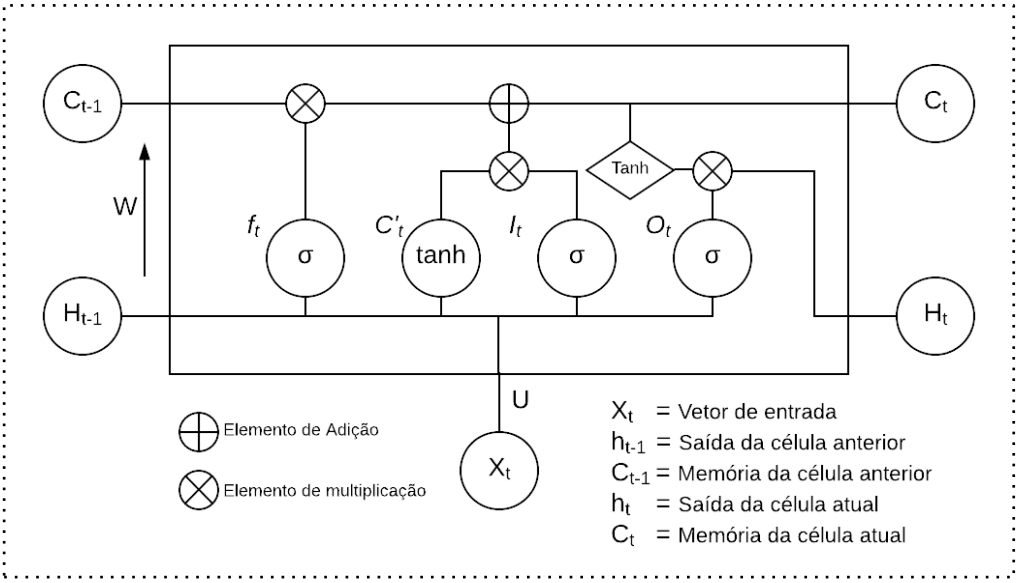

Figura 2.6: Detalhamento da célula LSTM

W e *U* são os vetores de pesos para f_t, C_t, I_t e O_t. A σ (sigmoid) e *tanh* (tangente hiperbólica) são funções de ativação que têm valores de 0 a 1 e de -1 a 1, respectivamente, como podemos ver na Figura 2.7[16].

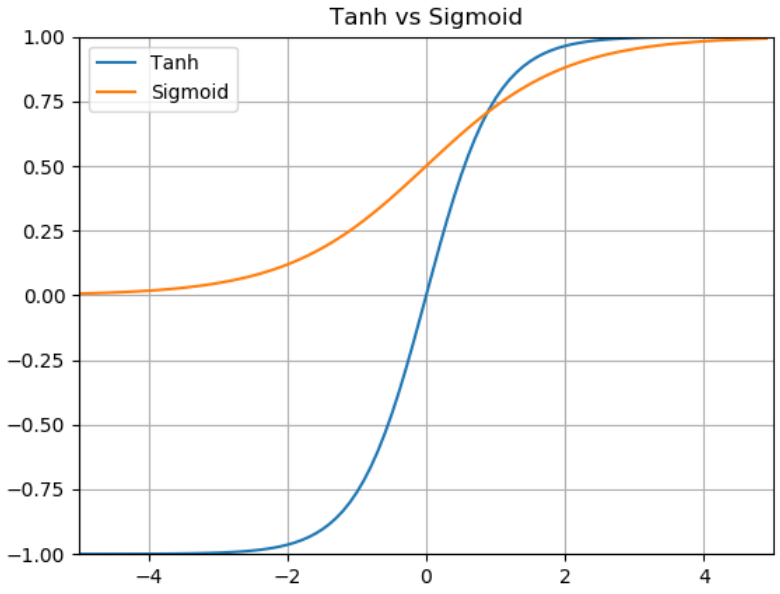

Figura 2.7: Funções Sigmoid (σ) e Tanh

A LSTM opera usando três portas: entrada, esquecimento e saída que são denotadas como i_t, f_t e O_t , respectivamente. A entrada X_t e o estado oculto anterior $H_{(t-1)}$ são usados por essas portas nas equações (2), (3) e (4):

$$f_t = \sigma(X_t * U_f + H_{t-1} * W_f), \quad (2)$$
$$I_t = \sigma(X_t * U_i + H_{t-1} * W_i), \quad (3)$$
$$f_t = \sigma(X_t * U_o + H_{t-1} * W_o), \quad (4)$$

Observe que as portas são dependentes de *H* e *X*. Dito isto, espera-se que o novo estado de memória (saída da célula) seja calculado como mostra a equação (5).

$$C_t = f_t * C_{t-1} + I_t * C_t', \quad (5)$$

A equação acima esquece algo do estado anterior, mas também acrescenta alguma entrada. Mas algumas informações a mais são necessárias para determinar o que vai como entrada da célula. Portanto, além de $I_{(t)}$, temos que calcular o que poderia possivelmente entrar no estado da célula (memória). Esta informação candidata está na equação (6):

$$C_t' = tanh(X_t * U_c + H_{t-1} * W_c), \quad (6)$$

E por fim, na equação (7) o novo estado é calculado, esse estado oculto agora é usado para calcular o que esquecer, a entrada e a saída da célula na próxima etapa.

$$H_t = O_t * tanh(C_t), \quad (7)$$

3 Metodologia

A técnica de previsão de séries temporais proposta neste trabalho pode ser dividida em duas fases, pré-processamento da entrada (Figura 3.1 e Figura 3.2) para preparar os dados a serem treinados ou testados e o processo de previsão em si (Figura 3.3). É importante ressaltar que ambos os fluxos possuem alguns componentes em comum. Como o intuito é a previsão de fluxo de caixa será aplicado aprendizado supervisionado. Cada componente é explicado em mais detalhes nas seguintes subseções.

3.1 Visão geral do processo

A seguir será apresentada uma série de imagens que descrevem o passo a passo, com abstração de detalhes para um entendimento geral do que foi desenvolvido neste trabalho, em seguida um detalhamento de cada processo efetuado.

A Figura 3.1 mostra o *dataset* sendo transformado de série temporal para dados estacionários realizando a diferença. Após isso, os dados são transformados para supervisionados e em seguida dividido em dados de treinamento e dados de teste.

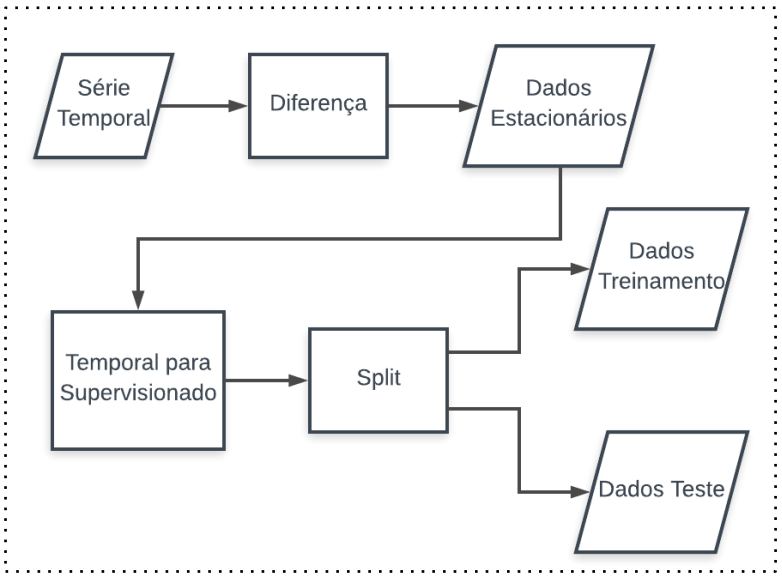

Figura 3.1: Diferenciação, supervisionamento e divisão

Na Figura 3.2 são trabalhados os dados de treinamento onde esse dado será normalizado, cada valor será convertido em uma faixa entre -1 e 1. O dado normalizado é usado para o treinamento da rede e é gerado um modelo onde este é usado para realizar a predição. Com o modelo e os dados de treinamento é possível realizar a predição para aperfeiçoar o modelo. Em uma das redes usadas, a LSTM, esse aperfeiçoamento é criado pela sua memória já citada anteriormente.

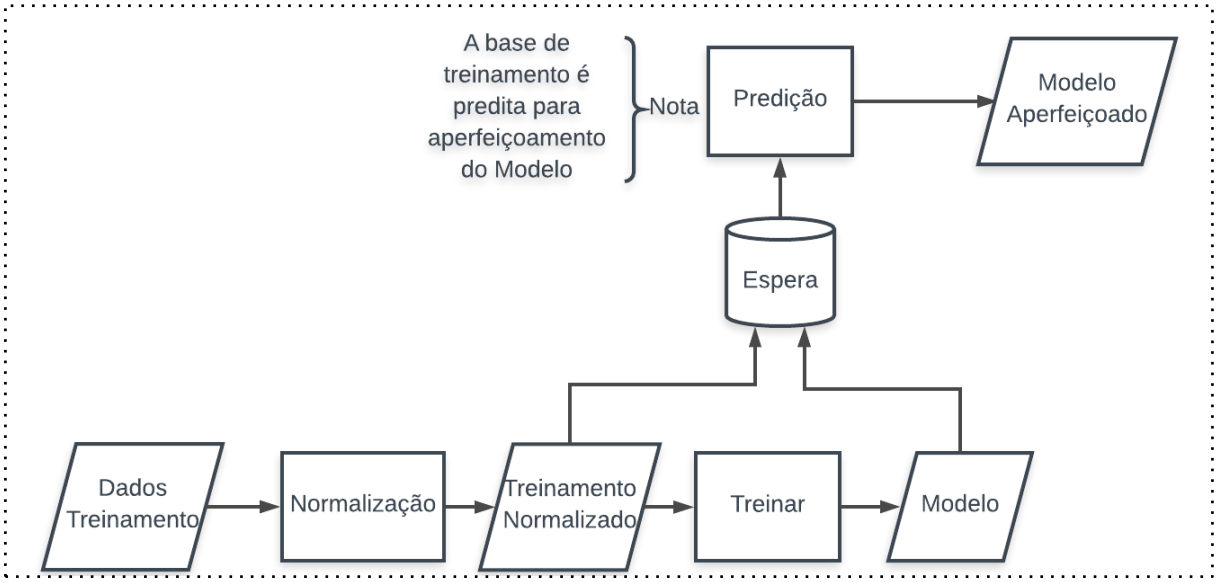

Figura 3.2: Normalização dos dados de treinamento, treinamento e aperfeiçoamento

Por fim, a Figura 3.3 mostra os dados de teste sendo normalizados da mesma forma como descrito acima, e com o modelo aperfeiçoado é realizada a previsão. Como será explicado com mais detalhes em seções futuras, a predição não é feita de única uma vez, o processo de predição recebe apenas um dado por vez, o oposto é feito no aperfeiçoamento, onde os dados de treinamento são preditos como um todo.

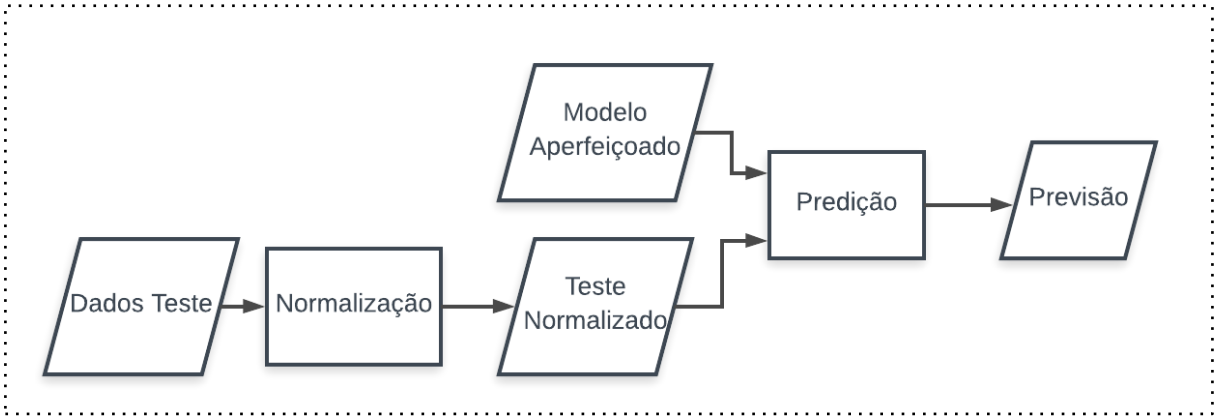

Figura 3.3: Normalização dos dados de teste e predição

3.2 Detalhando o processo

3.2.1 Dataset

A base de dados[17] utilizada neste trabalho contém diversas informações dentre elas:

- Nome do caixa
- Data da transação
- Número de retiradas
- Quantia total retirada
- Dia da semana

Focaremos em data da transação e quantidade total retirada. Uma amostra desses valores pode ser visualizada na Tabela 3.1. O *dataset* pode ser encontrado em um site onde ocorre a execução de diversas competições de ciência de dados e aprendizado de máquina chamado Kaggle[18].

O *dataset* possui informações sobre caixas automáticos (ATMs) do *City Union Bank* da Índia onde a moeda é a Rupia (Rs). A fim de não influenciar de qualquer forma o estudo os valores foram mantidos como os originais.

Tabela 3.1: Amostra do *dataset*

Data	Valor (Rs)
01/01	503400
02/01	268600
03/01	603900
04/01	541600
05/01	530000

Rupia (Rs) é equivalente a 0,05 Reais (R$)[1].

[1] Dado convertido em: https://themoneyconverter.com/INR/BRL.aspx. Acessado em 21 de maio de 2018.

3.2.2 Transformar Séries Temporais em Estacionárias

O conjunto de dados apresentados na seção anterior não é estacionário. Isso significa que há uma estrutura nos dados que depende do tempo. Uma série temporal é dita estacionária quando ela se desenvolve no tempo aleatoriamente ao redor de uma média constante, refletindo alguma forma de equilíbrio estável. Na prática, a maioria das séries que encontramos apresentam algum tipo de não estacionariedade, por exemplo, a tendência presente no conjunto de dados apresentado.

Dados estacionários são mais fáceis de modelar e muito provavelmente resultarão em previsões com resultados melhores. Se a série temporal não for estacionária, podemos transformá-la em estacionária com a seguinte técnica: diferenciação dos dados. Isto é, dada a série Z_t, criamos a nova série da equação (8).

$$Y_t = Z_t - Z_{t-1}, \qquad (8)$$

A Figura 3.4 mostra a comparação de uma série temporal e o resultado da aplicação da diferenciação. Podemos observar que a série diferenciada não tem o crescimento que a original possui e passa a variar em torno de um intervalo. Os dados diferenciados conterão um ponto a menos que os dados originais. Embora se possa diferenciar os dados mais que uma vez, uma diferenciação é geralmente suficiente. Pode-se aplicar essa transformação nas observações e depois retirá-la nas previsões para retornar os dados à escala original o que torna possível calcular uma pontuação de erro comparável.

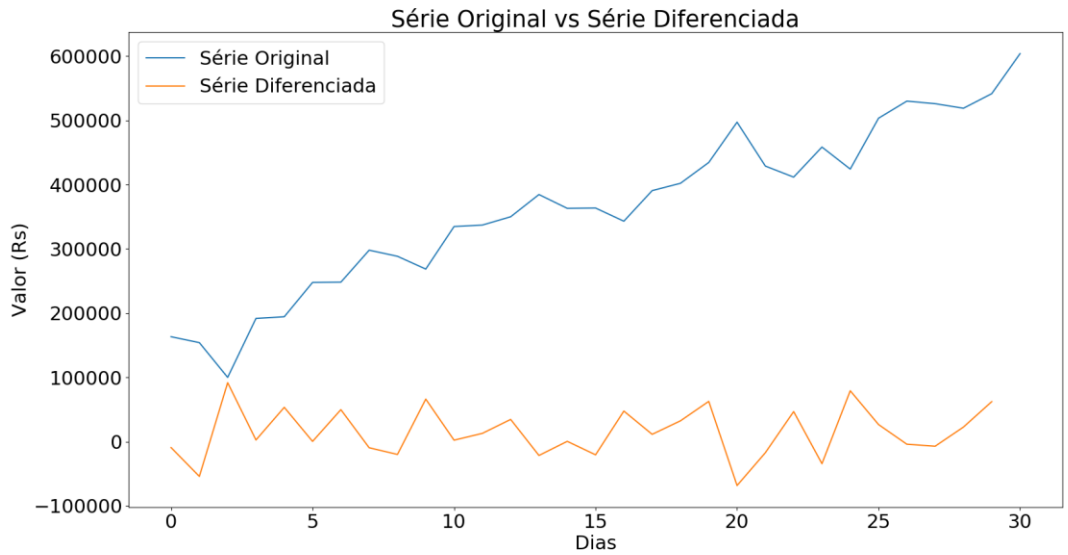

Figura 3.4: Série Original vs Série Diferenciada

Aplicando essa função aos dados da seção anterior obtemos o resultado da Tabela 3.2.

Tabela 3.2: Amostra do *dataset* quando aplicado a diferença

Índice	Valor(Rs)
1	-234800
2	335300
3	-62300
4	-11600

3.2.3 Aplicando aprendizado supervisionado

Os modelos LSTM e MLP assumem que seus dados são divididos em componentes de entrada *(X)* e saída *(y)*.

Para um problema de série temporal, podemos obter *X* observando os últimos intervalos de tempo *(t-n, t-n-1, …, t-2, t-1)* que se torna entrada do problema. E como saída, a observação do tempo atual *(t)*, ou seja *y*. Note que *n* é o número de passos. Para esse trabalho é usado *n* igual a 4 e igual a 1 com intuito de realizar uma comparação dos resultados da predição de um passo com a predição de múltiplos passos.

Podemos conseguir isso usando a função *shift()* da biblioteca Pandas, que pode empurrar todos os valores de uma série uma posição para baixo, para o caso de um deslocamento de 1 passo (n=1), que se tornarão as variáveis de entrada. Assim, a série temporal sem o deslocamento vem a ser a saída.

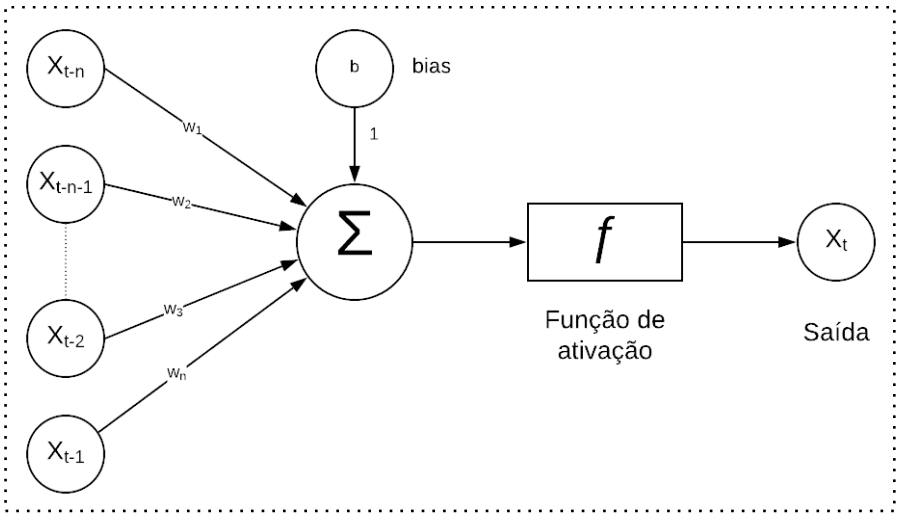

Figura 3.6: Exemplo de rede com aprendizado supervisionado

Na Figura 3.6 vemos um exemplo ilustrativo de uma rede de aprendizado supervisionado que recebe os valores dos intervalos de tempo anteriores (x_{t-n}, x_{t-n-1},... ,x_{t-2}, x_{t-1}) e tenta prever o valor de intervalo de tempo atual x_t.

Na Tabela 3.3 observamos como fica o *dataset* após a aplicação do supervisionamento de um passo.

Tabela 3.3: Amostra do *dataset* quando aplicado o supervisionamento

(t-1)	(t)
0	-234800
-234800	335300
335300	-62300
-62300	-11600

3.2.4 Normalização

Como outras redes neurais, as LSTM e MLP esperam que os dados estejam dentro da escala da função de ativação usada pela rede. A normalização é uma técnica geralmente aplicada como parte da preparação de dados para aprendizado de máquina. O objetivo da normalização é alterar os valores das colunas numéricas no conjunto de dados para usar uma escala comum, sem distorcer as diferenças nos intervalos de valores ou perder informações.

Os valores dos coeficientes de escala (mínimo e máximo) devem ser calculados no conjunto de dados de treinamento e aplicados para dimensionar o conjunto de dados de teste e as previsões. Isso evita contaminar o experimento com conhecimento do conjunto de dados de teste, o que pode dar ao modelo uma pequena vantagem.

É possível transformar o conjunto de dados no intervalo [-1, 1] usando a classe *MinMaxScaler* da biblioteca *sklearn*.

A Figura 3.7 mostra a comparação de uma série antes da normalização e depois da normalização. Vemos que apenas a escala mudou e a forma da série permanece a mesma.

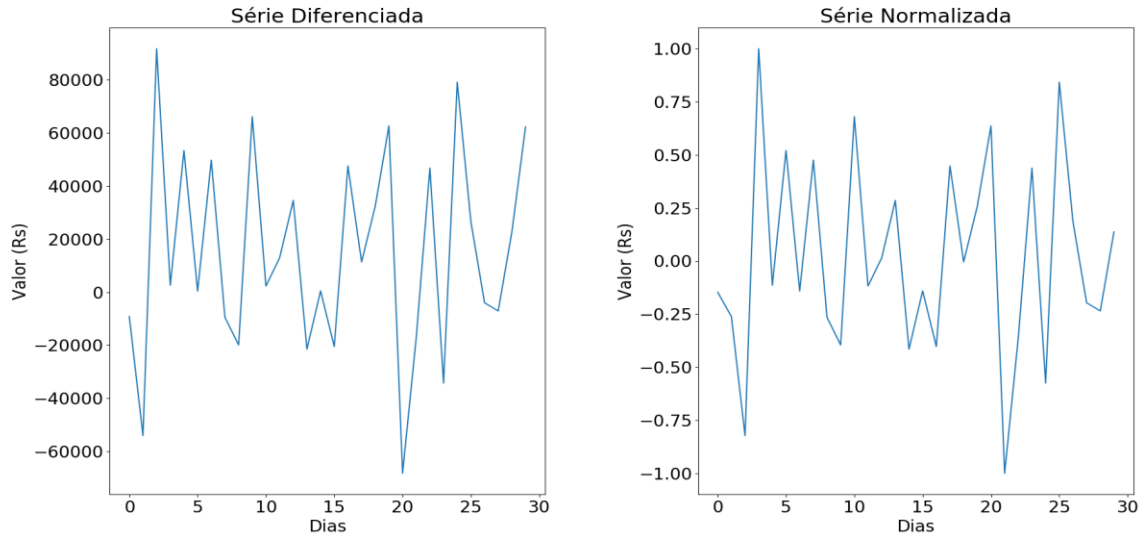

Figura 3.7: Série Diferenciada vs Série Normalizada

Aplicando a normalização ao retorno da seção anterior obtemos o resultado da Tabela 3.4.

Tabela 3.4: Amostra do *dataset* quando aplicado a normalização

(t-1)	(t)
-0,13628486	-1
-1	1
1	-0,39484301
-0,39484301	-0,21697948
-0,21697948	-0,5113138

3.2.5 Ajustando as redes

O ajuste da rede é responsável por gerar um modelo para que se possa realizar as previsões. Esse modelo é gerado a partir dos dados de treinamento configurando a rede com parâmetros. Para a configuração da LSTM é usado *batch_size* que define o número de amostras que serão propagadas pela rede.

Por exemplo, digamos que se tenham 1050 amostras de treinamento e que se deseja configurar *batch_size* igual a 100. O algoritmo obtém as primeiras 100 amostras (do 1º ao 100º) do conjunto de dados de treinamento e treina a rede. Em seguida, as outras 100 amostras (de 101 a 200) e treina a rede novamente. Para esse trabalho foi definido *batch_size*=1 para os números de amostras serem atualizados após cada passo de treinamento que é o padrão da biblioteca usada.

Isso possibilita a previsão de um passo, ou seja, um valor de cada vez. Uma vez que o modelo LSTM esteja ajustado aos dados de treinamento, ele pode ser usado para fazer previsões.

O ajuste da MLP é mais simples só usando a base de treinamento para gerar o modelo que vai ser usado para gerar as previsões. Após a aplicação do processo acima, uma parte do resultado pode ser visualizado na Tabela 3.5.

Tabela 3.5: Amostra do resultado da previsão

Data	Valor Esperado (Rs)	Valor Previsto (Rs)
20/04	624800	618390,99
21/04	440600	570182,11
22/04	643600	570263,80
23/04	468700	451594,62
24/04	324000	362245,80

3.3 Métodos para Análise

3.3.1 Erro percentual absoluto médio (MAPE)

O erro percentual absoluto médio (MAPE) é uma das medidas de precisão mais amplamente utilizada para avaliar previsões devido às suas vantagens de independência de escalas e interpretabilidade.

A maioria das pessoas está confortável pensando em termos percentuais, tornando o MAPE fácil de interpretar. Também pode transmitir informações quando não se sabe o volume de demanda do que se quer expor. Por exemplo, um funcionário dizer ao seu gerente, "estávamos com menos de 5% dos itens" é mais significativo do que dizer "estávamos com menos de 20 itens", se o gerente não sabe o volume típico de um item em estoque.

A fórmula para o MAPE está descrita na equação (9), onde n é o número de amostras, A_t é o valor esperado e F_t é o valor previsto.

$$MAPE = \frac{100\%}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|, \qquad (9)$$

3.3.2 Teste *Wilcoxon*

O Teste *Wilcoxon* é um exemplo de um teste não paramétrico ou de distribuição livre. É usado para testar a hipótese nula onde a diferença entre os pares segue uma distribuição simétrica em torno de zero (ou sem diferenças).

Para determinar se a diferença entre o valor esperado e o valor previsto é estatisticamente significante é necessário comparar o valor de p (ou probabilidade calculada) com o nível de significância. O nível de significância (denotado como α) de 0,05 é o mais usado. Se o valor de p for menor ou igual ao nível de significância ($p \leq \alpha$), a hipótese nula é rejeitada. Assim, é possível concluir que a diferença entre o valor esperado e o valor previsto é estatisticamente significativa. Se o valor de p for maior do que o nível de significância ($p > \alpha$), a hipótese nula não é rejeitada. Dessa forma, não há evidências suficientes para concluir que o valor esperado é significativamente diferente do valor previsto.

3.3.3 Média Móvel

É a média dos dados de uma série temporal de vários períodos consecutivos. Chamada de "móvel" porque é continuamente recalculada à medida que novos dados são disponibilizados, ela progride descartando o valor mais antigo e adicionando o valor mais recente. Por exemplo, a média móvel das vendas de seis meses pode ser calculada considerando a média das vendas de janeiro a junho, a média das vendas de fevereiro a julho, depois de março a agosto e assim por diante.

Médias móveis reduzem o efeito de variações temporárias nos dados, melhoram o ajuste dos dados a uma linha (um processo chamado de suavização) para mostrar a tendência dos dados mais claramente e destacam qualquer valor acima ou abaixo da tendência. Esse método vai ser utilizado para comparação com os resultados obtidos nos experimentos a fim de avaliar o desempenho.

4 Resultados e Análises

Nesta seção, são apresentados os resultados obtidos na aplicação dos algoritmos de predição LSTM e MLP. Os algoritmos foram executados usando o *dataset* apresentado na Seção 3.3.1. O *dataset* possui dados de 5 anos com 80% da base usada para treinamento e os 20% restantes para teste. Na Seção 4.1 são apresentados os resultados da LSTM. A Seção 4.2 é dedicada para os resultados da MLP. E por último, na Seção 4.3 é feita a análise usando os métodos citados na Seção 3.3.

4.1 Resultados LSTM

O experimento foi executado para dois ATMs chamados *Airport* e *Big Street*, com previsão de um passo (t-1) e múltiplos passos (t-4). Em seguida os gráficos comparam os valores reais de saque nos ATMs (linha azul) e os valores previsto pela LSTM (linha laranja). As Figuras servirão para uma análise visual dos resultados, na seção 4.3 vão ser feitas as análises numéricas com os erros MAPE e teste de *Wilcoxon.*

Na Figura 4.1 estão os resultados do Airport para a previsão de um passo.

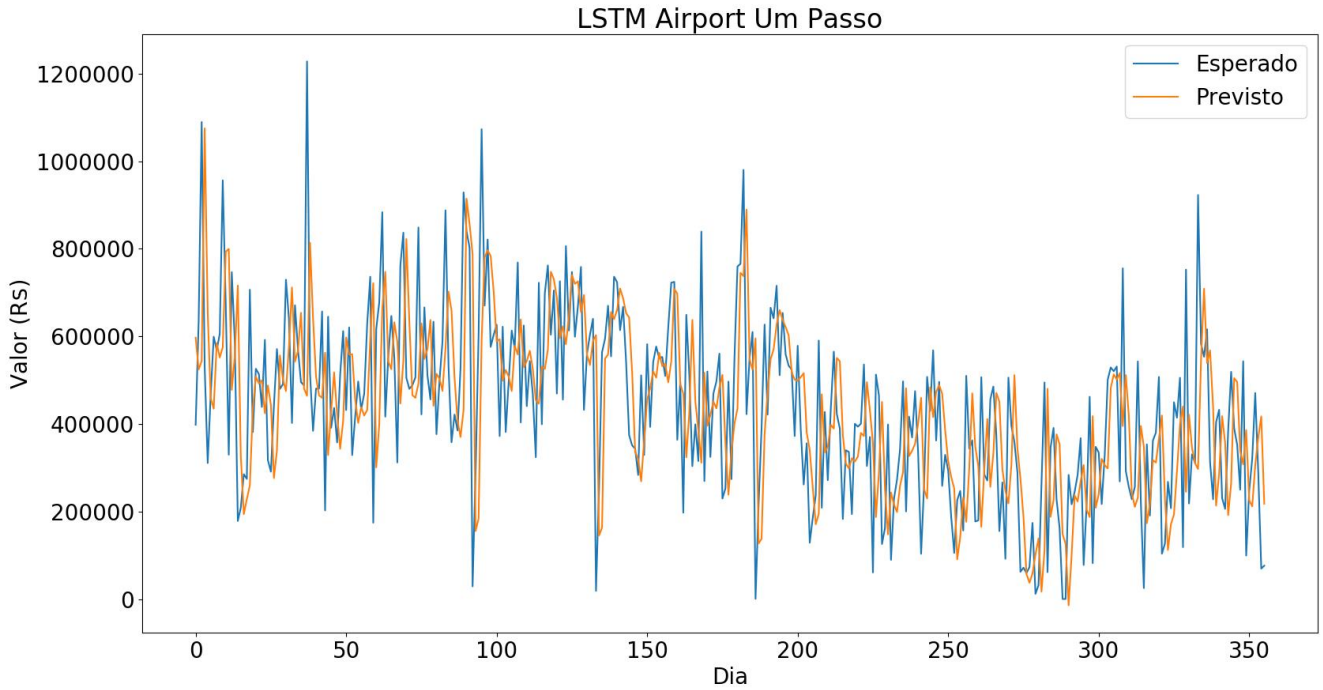

Figura 4.1: ATM Airport valor esperado vs. previsto

Na Figura 4.2 os resultados do Airport para a previsão de múltiplos passos.

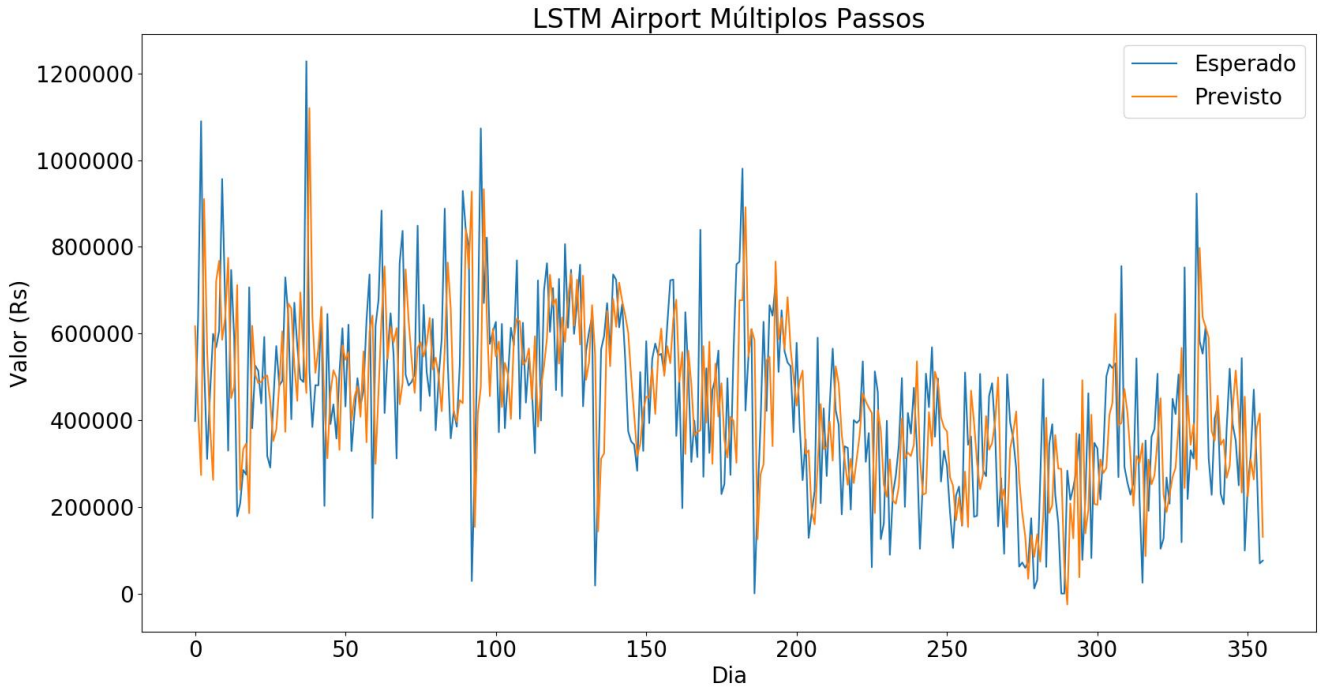

Figura 4.2: ATM Airport valor esperado vs. previsto

Na Figura 4.3 os resultados do Big Street para a previsão de um passo.

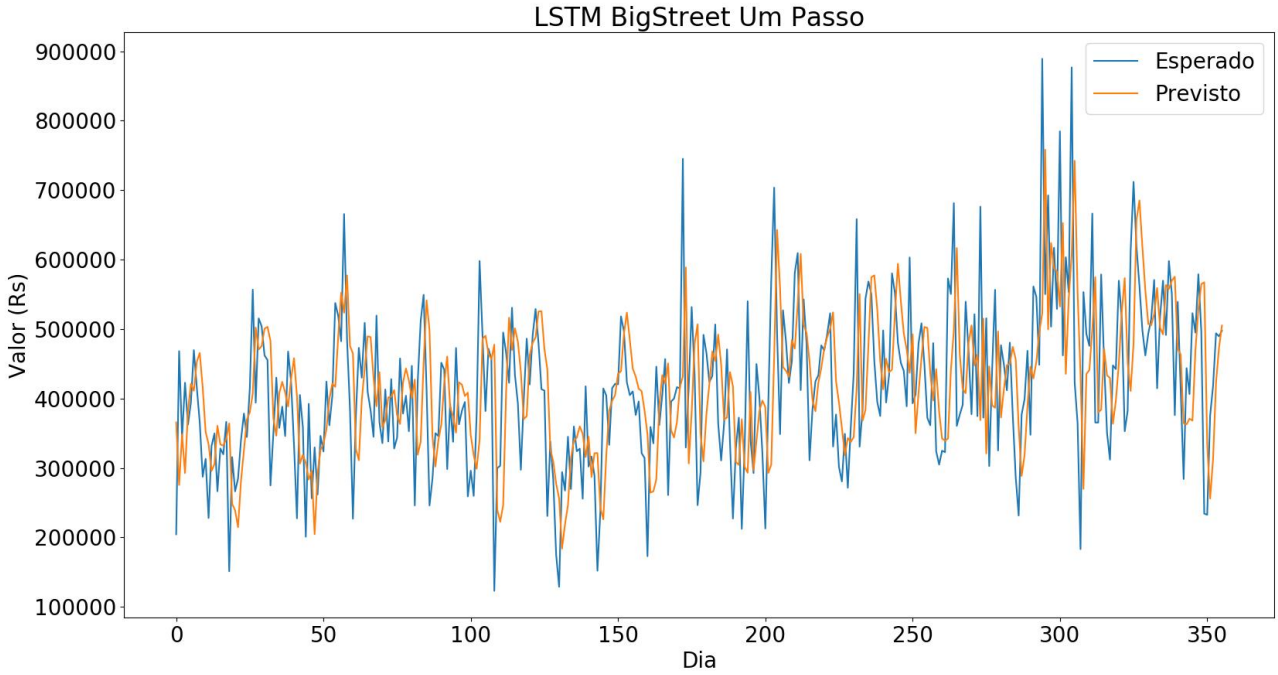

Figura 4.3: ATM Big Street valor esperado vs. previsto

Na Figura 4.4 os resultados do Big Street para a previsão de múltiplos passos.

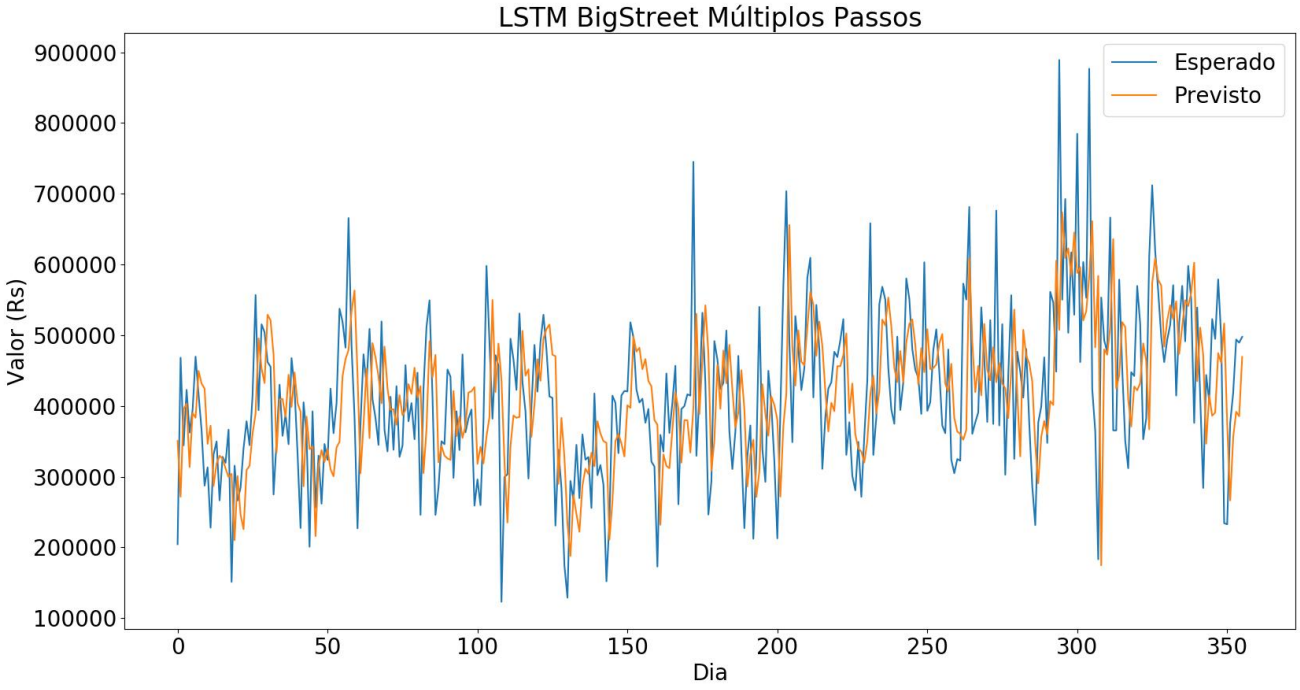

Figura 4.4: ATM Big Street valor esperado vs. previsto

Observamos que para o ATM *Airport* a predição com múltiplos passos se aproximou mais visualmente dos valores esperados acompanhando mais os valores de pico e de vale, diferente da predição com um único passo que tem seus picos e vale mais suavizados.

Para o ATM *Big Street* o oposto ocorre, com a previsão de um passo obteve-se uma maior aproximação dos valores de pico e vale, mas com a previsão de múltiplos passos ocorre a suavização, apesar de menor do que a previsão de um passo do ATM *Airport*.

4.1 Resultados MLP

Como na seção anterior foram usados os ATMs Airport e Big Street, com previsão de um passo (t-1) e múltiplos passos (t-4). Em seguida os grafos comparam os valores reais de saque nos ATMs (linha azul) e os valores previsto pela MLP (linha laranja). As Figuras serviram para uma análise visual dos resultados, na próxima seção vão ser feitas as análises numéricas com os erros MAPE e teste de *Wilcoxon.*

Na Figura 4.5 os resultados do Airport para a previsão de um passo.

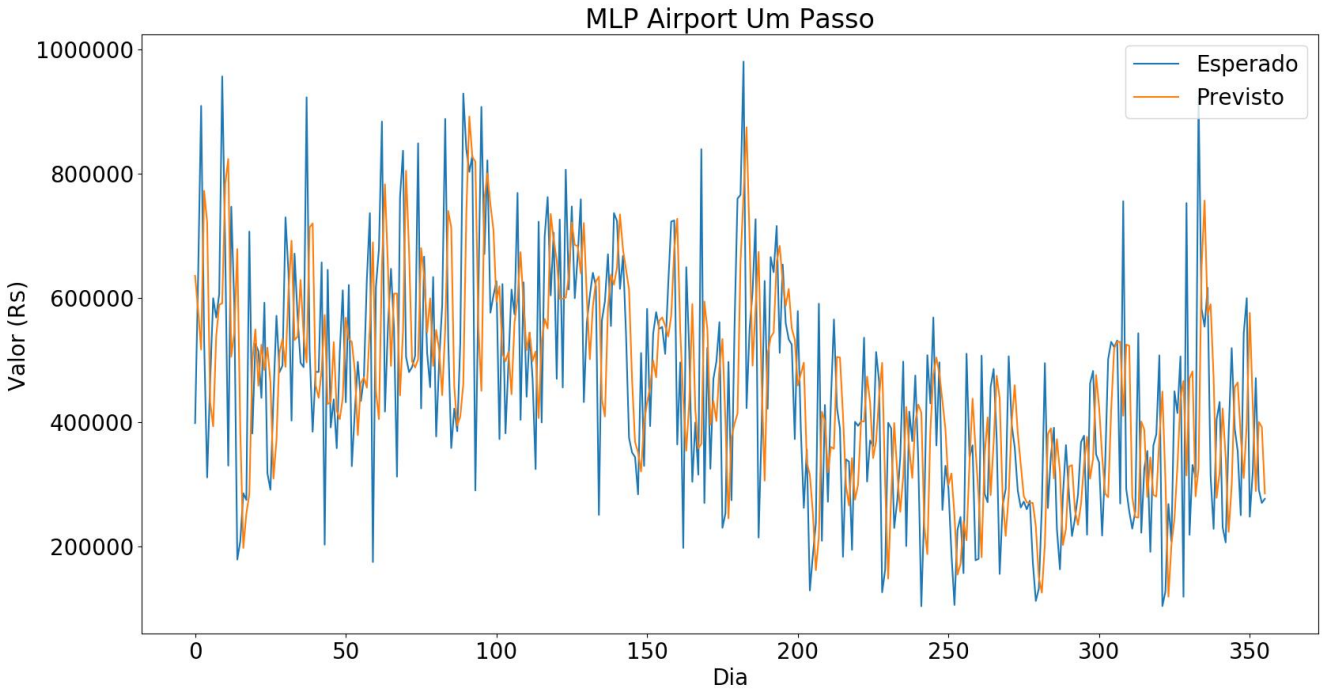

Figura 4.5: ATM Airport valor esperado vs. previsto

Na Figura 4.6 os resultados do Airport para a previsão de múltiplos passos.

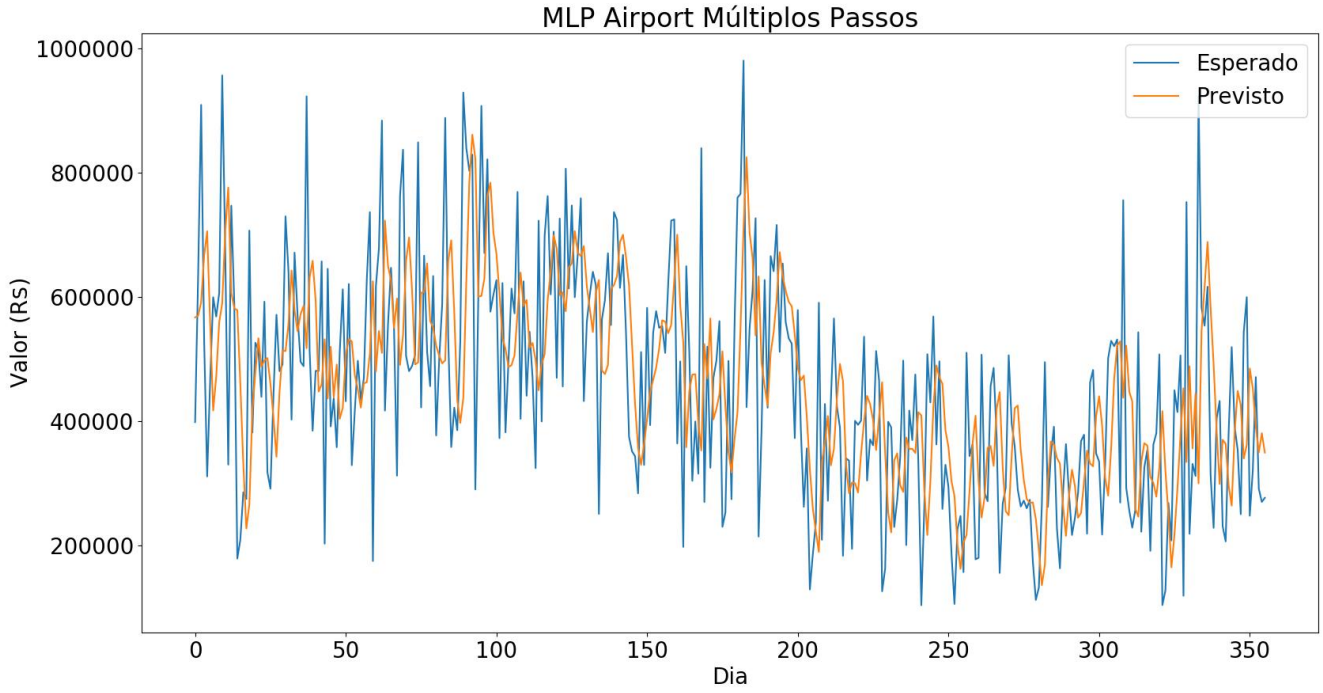

Figura 4.6: ATM Airport valor esperado vs. previsto

Na Figura 4.7 os resultados do Big Street para a previsão de um passo.

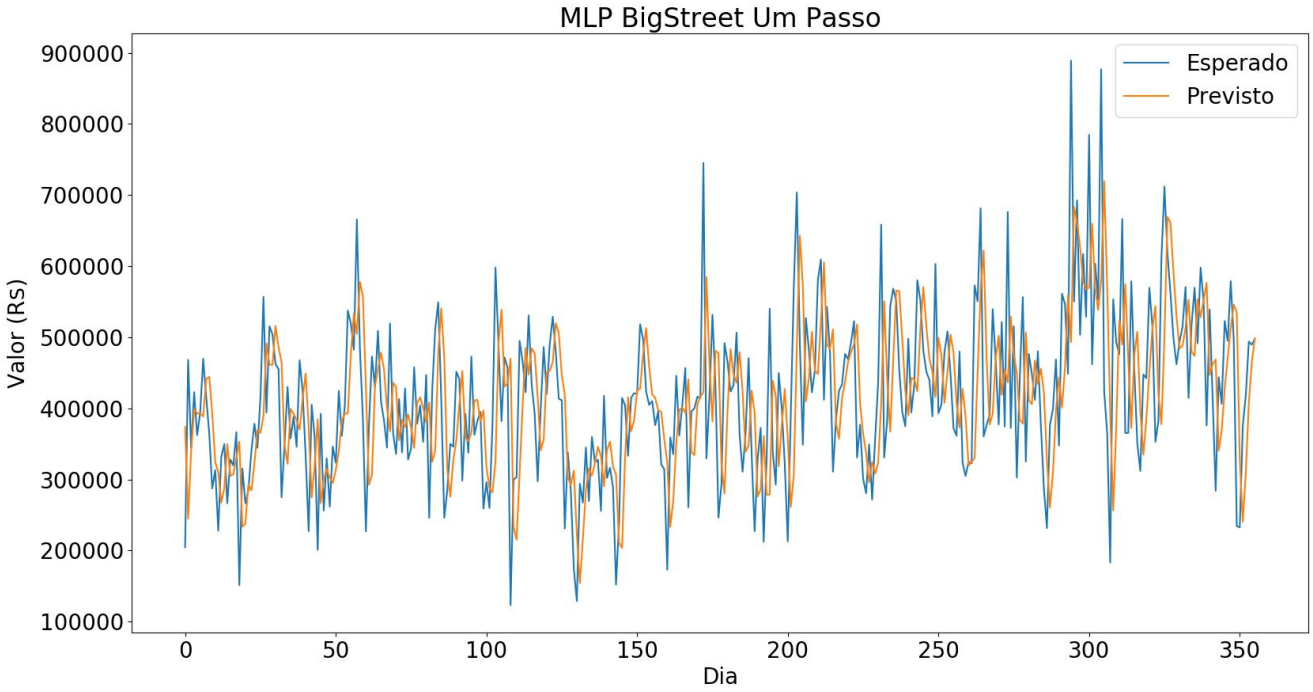

Figura 4.7: ATM Big Street valor esperado vs. previsto

Na Figura 4.8 os resultados do Big Street para a previsão de múltiplos passos.

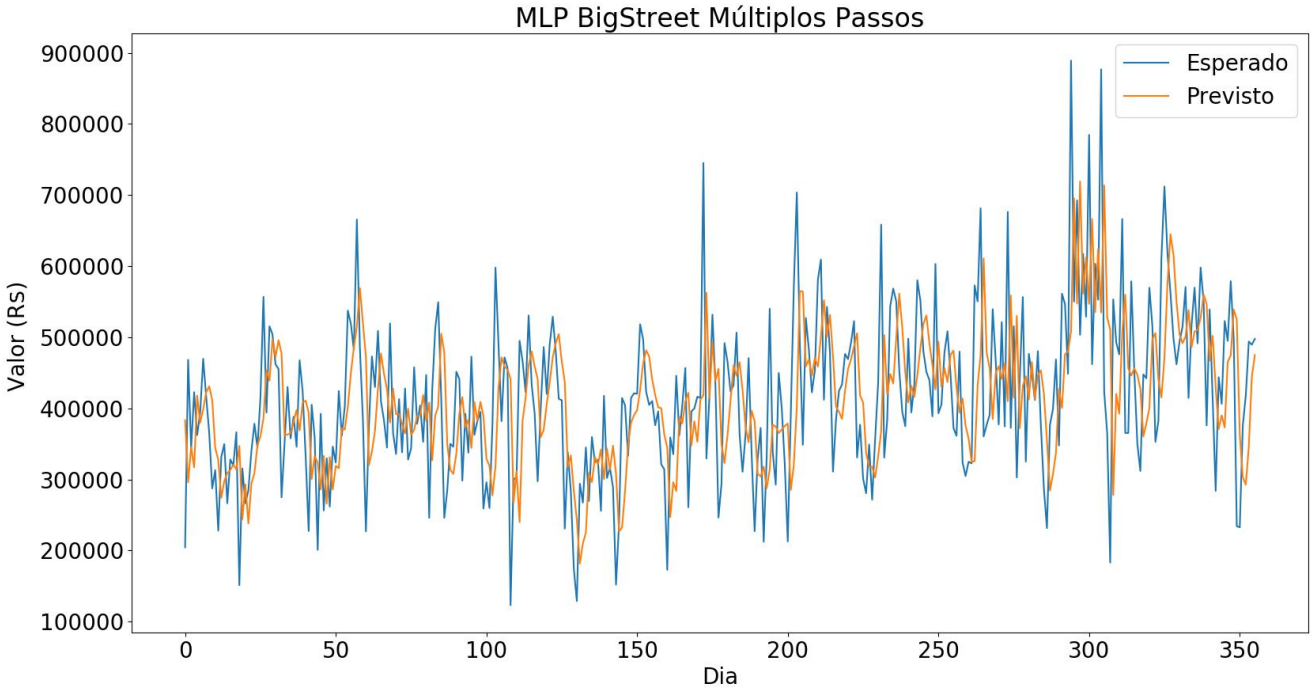

Figura 4.8: ATM Big Street valor esperado vs. previsto

Diferente da LSTM o comportamento da MLP em relação aos valores de picos e vale em ambas previsões de um passo para os dois ATMs foram visualmente melhor e a suavização ocorre na previsão de múltiplos passos.

4.3 Análises

Para a análise dos resultados foram utilizados: MAPE com o objetivo de mensurar o erro que os modelos obtiveram, o teste *Wilcoxon* para avaliar se o erro obtido tem significância estatística e por último foi adicionado uma margem para erro que vai servir para avaliar se os valores previstos estão dentro de um intervalo aceitável.

A Tabela 4.1 compara os resultados obtidos com o cálculo do MAPE. Nas últimas duas linhas da tabela encontramos o MAPE da média móvel. Note que o MAPE para a média móvel é calculado apenas duas vezes, uma para cada ATM.

Vemos que ambos os métodos tiveram um melhor desempenho com o *dataset* do ATM *Big Street*, com o LSTM saindo-se um pouco melhor. Já para o ATM *Airport* o desempenho foi muito similar.

Tabela 4.1: MAPE - Mean absolute percentage error

Cenário	**MAPE**
LSTM Airport um passo	31,972%
LSTM Airport múltiplos passos	36,231%
LSTM Big Street um passo	17,773%
LSTM Big Street múltiplos passos	18,110%
MLP Airport um passo	34,919%
MLP Airport múltiplos passos	33,386%
MLP Big Street um passo	19,437%
MLP Big Street múltiplos passos	18,808%
Média Móvel Airport	60,623%
Média Móvel Big Street	43,134%

A Tabela 4.2 mostra os p-*valores* do cálculo do teste *Wilcoxon*. Os parâmetros do teste foram 0,05 para o nível de significância e o bicaudal pois as diferenças entre os esperados e previstos resultam com valores positivos e negativos.

Tabela 4.2: p-valores do teste *Wilcoxon*

Cenário	**P-valores**
LSTM Airport um passo	0,359638
LSTM Airport múltiplos passos	0,850596
LSTM Big Street um passo	0,601063
LSTM Big Street múltiplos passos	0,326642
MLP Airport um passo	0,449030
MLP Airport múltiplos passos	0,451810
MLP Big Street um passo	0,995073
MLP Big Street múltiplos passos	0,627461

Vemos então que as previsões não possuem diferença estatisticamente significativa quando comparadas com os valores reais, pois o valor de p é maior que o nível de significância não rejeitando a hipótese nula, descrita na Seção 3.3.2.

5 Considerações Finais

O objetivo principal deste trabalho é analisar a aplicação dos algoritmos de rede neurais da aprendizagem de máquina para fornecer uma previsão de fluxo de caixa em ATMs. Foram usadas as redes LSTM e MLP backpropagation com um *dataset* que possui informações de dois ATMs do *City Union Bank* da índia, os ATMs denominados de *Airport* e *BigStreet*.

Para tal, foram implementadas etapas de pré-processamento da base de dados, experimentação, treinamento e predição. Os pré-processamentos foram a diferenciação dos dados da base para transformá-la em estacionária, normalização e a transformação em aprendizagem supervisionada.

Na análise dos resultados, observamos que ambos os algoritmos obtiveram bom desempenho com os dados do ATM *BigStreet*, apresentando um MAPE abaixo de 20%. Com os dados do ATM *Airport*, porém, o MAPE foi um pouco de maior que 30%. Para fins comparativos, foi calculado o MAPE para a média móvel, sendo ela 43% para o ATM *BigStreet* e 60% para o ATM *Airport*. O teste de *Wilcoxon* também foi aplicado nos resultados e como foi apresentado a diferença dos valores das previsões e os dados esperados não se mostrou estatisticamente significante. Com isso, podemos concluir que as abordagens de tratamento dos dados e experimentação foram eficientes em lidar com as bases de dados utilizadas. E os dois algoritmos experimentados, a LSTM, e a MLP, foram capazes de performar bem sobre o *dataset BigStreet*, mostrando que para um grupo de informações diferentes os resultados podem variar significativamente apenas variando a fontes dos dados.

5.1 Trabalhos futuros

Alguns possíveis trabalhos futuros são:

- Usar bases de dados que os valores sejam em Real ou em Dólar.
- Variar com mais intensidade os parâmetros fornecidos pelas bibliotecas que possuem as implementações da MLP e da LSTM, scikit-learn e Keras, respectivamente.
- Atualizar o modelo LSTM. O modelo pode ser atualizado em cada etapa da validação.
- Usar mais entradas para as redes como o dia da semana e se o dia é feriado local.

Referências

[1] ARORA, N.; SAINI, J. K. R. **Approximating Methodology: Managing Cash in Automated Teller Machines using Fuzzy ARTMAP Network**. International Journal of Enhanced Research in Science Technology & Engineering, Vol. 3, No. 2, 2014.

[2] ATM Marketplace. **Global ATM installed base to reach 4M by 2021**. https://atmmarketplace.com/news/global-atm-installed-base-to-reach-4m-by-2021/. Acesso em: 28 março 2018.

[3] SIMUTIS, R. et al. **Optimization of cash management for ATM network**. Information Technology And Control, Vol. 36, No. 1A, 2007.

[4] Investopedia. **The Banking System: Commercial Banking - How Banks Make Money**. https://www.investopedia.com/university/banking-system/banking-system3.asp. Acesso em: 28 Março 2018.

[5] Encyclopedia Britannica. **Machine learning**. https://global.britannica.com/technology/machine-learning. Acesso em: 28 março 2018.

[6] SIMUTIS, R., DILIJONAS, D., BASTINA, L.: **Cash demand forecasting for ATM using neural networks and support vector regression algorithms**.
Proc. of the 20th Euro Mini Conf. on Continuous Optimization and Knowledge-Based Technologies, 2008.

[7] CARPENTER, G. et al. **Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps**, IEEE Transactions on Neural Networks, Vol. 3, No. 5, 1992.

[8] ACUÑA, G., RAMIREZ, C., CURILEM, M., **Comparing NARX and NARMAX models using ANN and SVM for cash demand forecasting for ATM**, The 2012 International Joint Conference on Neural Networks (IJCNN), 2012

[9] NILSSON, N.: **Introduction to Machine Learning**. [S.l.: s.n.], 1996.

[10] MITCHELL, T.: **Machine Learning**. McGraw-Hill, 1997

[11] ROSENBLATT, F.: **The Perceptron: A Probabilistic Model For Information Storage And Organization In The Brain**. Psychological Review Vol. 65, No. 6, 1958.

[12] MINSKY, M., PAPERT, S.: **Perceptrons: an introduction to computational geometry**. The MIT Press, 1969

[13] RUMELHART, D., HINTON, G., WILLIAMS, R.: **Learning representations by back-propagating errors**. Nature Publishing Group, 1986

[14] Christopher Olah. **Understanding LSTM Networks**. http://colah.github.io/posts/2015-08-Understanding-LSTMs/. Acesso em: 05 junho 2018.

[15] HOCHREITER, S., SCHMIDHUBER, J.: **Long Short-Term Memory**. Neural Computation, Vol 9, No. 8, 1997.

[16] Towards Data Science. **Activation Functions: Neural Networks**. https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6 . Acesso em: 04 Junho 2018.

[17] Kaggle .**ATM Transaction Data of City Union Bank**. https://www.kaggle.com/nitsbat/atm-transaction-data-of-city-bank. Acesso em: 20 Março 2018.

[18] TechCrunch. **Google is acquiring data science community Kaggle**. https://techcrunch.com/2017/03/07/google-is-acquiring-data-science-community-kaggle/. Acesso em: 28 março 2018.