ADRIANO DENNANNI
RICARDO NAGANO
THIAGO LIRA

NET.MAP - SISTEMA DE POSICIONAMENTO INDOOR

São Paulo
2016

ADRIANO DENNANNI
RICARDO NAGANO
THIAGO LIRA

NET.MAP - SISTEMA DE POSICIONAMENTO INDOOR

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro Eletricista com ênfase em Computação.

São Paulo
2016

ADRIANO DENNANNI
RICARDO NAGANO
THIAGO LIRA

NET.MAP - SISTEMA DE POSICIONAMENTO INDOOR

Trabalho apresentado à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro Eletricista com ênfase em Computação.

Área de Concentração:
Engenharia de Computação

Orientador:
Prof. Dr. Reginaldo Arakaki

Co-orientador:
Eng. Marcelo Pita

São Paulo
2016

AGRADECIMENTOS

"Anything one man can imagine, other men can make real"

-- Jules Verne

"Enquanto sentir vontade de competir, buscar desafios e correr atrás de torneios, vou jogar"

-- Gustavo Kuerten

"Not all those who wander are lost"

-- J. R. R. Tolkien

RESUMO

Mapas físicos tornam-se cada vez menos utilizados com o desenvolvimento progressivo de sistemas de posicionamiento cada vez melhores. O sistema americano GPS é possivelmente o mais utilizado, sendo que ele possibilita qualquer um ter informações sobre sua localização, dando apoio, por exemplo, à praticantes de trilhas e acampamentos, principalmente em casos de emergência. Porém, em ambientes fechados, as ondas eletromagnéticas utilizadas pelos satélites sofrem atenuações e interferências devidos aos materiais de construção, e assim o sistema perde precisão e não funciona com toda a precisão esperada. Como uma alternativa para esta dificuldade, procurou-se desenvolver um sistema, que consegue obter a posição do usuário em um ambiente fechado com precisão, sendo usado para isso técnicas de machine learning, aliadas com dados obtidos de redes em fio já instaladas no local. O sistema consistirá de um servidor central, onde serão enviados os dados e os mesmos serão processados. Os dados serão coletados por meio de um aplicativo de Android, este possuirá duas versões. A versão usuário usará os dados do servidor para localizar o usuário, a versão administrador irá coletar dados novos para serem usados em futuras medições.

Palavras-Chave – Localização Indoor, Wi-Fi, Machine Learning.

ABSTRACT

Physical maps are becoming each day less used due to constant evolution of positioning systems, better each day as well. The American system GPS probably is the most used and the most famous. It allows everyone to have their location information, giving support to hikers and campers, specially in emergency situations. On the other hand, in indoor environments, electromagnetic waves used by the satellites suffer with interference and mitigations and the systems loses precision and does not work as expected. As an alternative for this difficulty, it was developed a system that can locate the user position in an indoor environment with precision, using machine learning algorithms and data of wireless signals collected from the networks already existing on the place. The system consists on a main server that will receive the data and process it. The data will be collected with a Android app that will have two versions. The user version will use the server data to locate the user. The admin version will collect new data to be user on future measures.

Palavras-Chave – Indoor Location, Wi-Fi, Machine Learning.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 Objetivo

O objetivo deste projeto é desenvolver um conjunto de ferramantas que possibilitem o mapeamento e a indentificação de áreas dentro de ambientes fechados. Estas ferramentas serão utilizadas em dispositivos móveis, possibilitando que os usuários possam se localizar em locais fechados. Para chegar a esse objetivo, um sistema de Machine Learning utilizará os valores das potências das redes Wi-Fi presentes nos arredores para aprender a mapear diversas zonas no ambiente.

1.2 Motivação

Com a modernização das tecnologias de telefonia móvel torna-se cada vez maior o número de pessoas com acesso à *Internet*, através de tecnologias como *Wi-fi*, 3G e 4G. Essas formas de acesso à rede fornecem informações a provedores sobre o usuário a todo momento,como o conteúdo acessado por seus navegadores ou aplicativos e informações sobre posição e deslocamento. Dados de localização por si possuem pouco valor, mas quando aliados a outros conteúdos, é possível fornecer conteúdo personalizado em tempo real, reativo ao ambiente, passando a oferecer valor real à empresas e entidades.

Sistemas de posicionamento por satélite como GPS conseguem localizar um dispositivo na Terra com uma precisão na casa dos centímetros em ambientes abertos. Porém, o mesmo não ocorre em lugares fechados, como residências e edifícios. Isso ocorre devido à atenuação dos sinais dos satélites causada pelas paredes e tetos das estruturas. Tendo em vista o crescimento das cidades e o consequente aumento no número de construções, as pessoas cada vez passam mais tempo em ambientes fechados. A necessidade de serviços de localização *indoor* tem se tornado cada vez mais evidente.

Respondendo a essa necessidade, surgiram alternativas para o posicionamento em ambientes fechado, tais como o emprego de *tags RFID* (*Radio Frequency Identification*) e do

Bluetooth. Tendo em vista este cenário e as condições tecnológicas atuais, este projeto procura apresentar uma solução alternativa para localização de pessoas em ambientes fechados, como shoppings e eventos em galpões, sem ter que investir altos valores em infraestrutura. Para tal, será utilizada a tecnologia *Wi-fi* combinada a técnicas de *machine learning*.

Esta abordagem se mostra interessante ao ponto de que sua implementação não necessita de configurações particulares nas redes ao redor, uma vez que se baseia em leituras feitas pelo aparelho móvel e no processamento dos dados feitos em um servidor em nuvem. Tampouco será necessário se conectar a uma desses redes *Wi-Fi* no ambiente.

1.3 Justificativa

Levando em conta a falta de alternativas práticas para sistemas de posicionamento *indoor*, o *net.map* se mostra ideal para suprir essa demanda. O sistema pode ser utilizado por museus (para tornar a experiência mais interativa) ou por *Shopping Centers* (para sugerir produtos diferentes de acordo com a localização do cliente). Esses dois exemplos de ambientes são típicamente instalados em ambientes fechados, onde o GPS não funciona bem, e como consequência o *net.map* pode preencher essa lacuna funcionando como o sistema de localização padrão para esses lugares.

1.4 Organização

2 ASPECTOS CONCEITUAIS

Grande parte da pesquisa e fundamenta tea feita sobre *machine learning* foi feita no livro "An Introduction to Statistical Learning" [**?**] e no curso online ministrado por Andrew Ng [**?**]. A seguir sxplicados detalhadamente alguns pontos teos importantes usados no trabalho.

2.1 Bias vs. Variia: Um tradeoff

Um dilema comum que se aparece ao usar *Machine Learning* e *Bias* contra Variia. *Bias* edida do erro do modelo treinado e da realidade que nntamos modelar. Variia rro decorrente de pequenas flutuas nos pontos de treino, ou seja, diversos modelos treinados com pequenas mudan dos pontos de treino irerar erros drasticamente diferentes. Uma maneira comum de generalizar o erro de modelos de *Machine Learning* dividi-lo em uma parcela de erro proveniente de *bias*, outra de variia e uma terceira parcela de ru com ma nula. Esse dilema aparece pela contradi clara entre desejar reduzir *bias* e variia ao mesmo tempo. Um modelo com baixo *bias* is complexo, (e.g. um polinmio de ordem alta) de modo que a curva representa de maneira mais pra os pontos de treino, por isso tambcaba por fazer que o ru desse conjunto particular de pontos de treino seja capturado pelo modelo, o que n desejado, visto que queremos que o modelo se adeque a *qualquer* conjunto de pontos, npenas os que foram usados para treina-lo. A solu poderia ser diminuir ent complexidade do modelo, mas isso invariavelmente aumenta nosso erro de *bias*, por mais que diminuia o de variia.

2.2 Algoritmos de Machine Learning

2.2.1 KNN

O Algoritmo KNN busca criar regies de classifica no espaelo voto majorito dos K pontos de treino mais pros, sendo K um partro do algoritmo. Na imagem a seguir podemos ver essa classifica em duas dimenses, supondo que os pontos roxos representam uma classe, e

os verdes, outra. A imagem da esquerda representa a sa do algoritmo para um valor de K escolhido pequeno, e a da direta, para um valor grande K, com o risto de *overfit*. Pode-se reparar que com um K maior, um ponto solito que pode representar um erro dos dados de treino norado por estar cercado de pontos que representam outra classe.

Figura 1: Algoritmo KNN aplicado com diferentes valores de K

2.2.2 rvore de Decis

Em sua forma mais simples, ores de decisma classe de algoritmos que buscam (no caso de classifica) achar a classe de um ponto de treino testando intervalos de suas *features*. Vejamos o exemplo de uma ore treinada a seguir:

Figura 2: rvore de Decisara prever a sobrevivia de passageiros do Titanic

Essa ore busca prever se um determinado passageiro do Titanic sobreviveu ou no acidente testando diversas caractericas desse indivo. Por exemplo, podemos ver que caso o sexo do passageiro seja feminino, ela tem uma alta chance de ter sobrevivido. Caso contro, jo testadas outras duas clulas referentes a idade ao nmero de familiares tamb bordo do navio. Por constru, temos um *trade-off* para ores, podendo trocar **legibilidade** por **precis** (menos *bias*). ossl construir ores mais complexas que poder sua facilidade de interpreta por uma pessoa, porodere adequar melhor ao dados treinados, e sa ideia que plorada por algoritmos de *ensemble learning* como *Random Forests* ou *Adaboost*, que serxplicados em outra parte desse documento, mas que basicamente combinam o poder de diversas ores treinadas iterativamente, para aumentar seu poder de classifica.

2.2.3 Support Vector Machines

Essa classe de algoritmos busca, para um conjunto de dados N-dimensionais, encontrar um hiperplano que separa o espaos dados em regies de classifica. Vejamos um exemplo simples a seguir:

Figura 3: SVM aplicado para classificar dados linearmente separis

A reta na imagem foi encontrada de modo a separar o espae classifica da classe "azul"do espae classifica da classe "vermelha". A linha pontilhada representa a margem de classifica, o algoritmo busca chegar a um hiperplano que tenha ainda uma distia dos pontos mais pros da

regie separa. A seguir, um exemplo mais complexo usando um *kernel* para poder classificar dados ninearmente separis:

Figura 4: SVM aplicado para classificar dados de maneira ninear

3 ESPECIFICA

3.1 Requisitos

3.1.1 Requisitos Funcionais

- Enviar dados sobre a intensidade dos sinais de *Wi-fi* ao servidor.

- Retornar a posi do usuo baseado nos dados enviados.

- Disponibilizar uma API para ser usada por outras aplicas.

3.1.2 Requisitos NFuncionais

- O sistema deve ser transparente ao usuo.

3.2 Pontos de Vista

3.2.1 Ponto de Vista da Empresa

Este projeto deveriar um sistema de localiza precisa em ambientes fechados, utilizando como partros as intensidades de sinais redes sem fio pras. O resultado deste projeto incluiras aplicas:

Aplicativo para o celular com dois mos utilizados para capturar informas das redes sem fio pras e envias servidor, ale receber resultados deste servidor e exibi-los para o usuo;

Aplica em nuvem que receber dados dos celulares e responderm a localiza do usuo.

Para tal sereitas anses estaticas do comportamento e flutua do sinal de *Wi-fi* em diversas medis e com aparelhos de celular diferentes e posteriormente, estes dados sernalisados por um algoritmo de inteligia artificial baseado em *machine learning*, para que a localiza seja definida.

3.2.2 Ponto de Vista da Informa

As informas processadas pelo sistema seguirois fluxos. No primeiro as intensidades dos sinais provenientes dos pontos de acesso de redes sem fio serapturadas pelo sensor do celular e registradas pelo mo de aquisi de dados do aplicativo. Em seguida, estas informas sernviadas para o servidor para que recebam o devido tratamento.

Apero de treinamento do sistema o segundo fluxo serssl. Nele apispositivo m enviar os sinais capturados por um usuo regular, o servidor deve executar seu algoritmo de *machine learning* e retornar si que o usuo se encontra dentro do pro previamente estabelecido, usando como base estes dados enviados.

A troca de dados em ambos os fluxos serita usando o formato JSON, este formato ito utilizado principalmente quando cessa a implementa de tabelas de bancos de dados por conta de sua simplicidade para manipular as informas representadas.

3.2.3 Ponto de Vista da Computa

O sistema sermposto por duas partes funcionais, a primeira plicativo desenvolvido para dispositivos Android, ele sersponsl pela captura dos dados, em seguida ele organizar informas coletas no formato JSON e finalmente realizarransmissor meio da rede al o dispositivo estiver conectado, uma rede 3G/4G ou pela conex uma rede *Wi-fi* disponl.

A segunda parte ervidor em nuvem, que deverocessar os dados executando o algoritmo de *machine learning* e em seguida disponibilizar resultados obtidos para posterior consulta feita quando forem requisitadas informas a respeito do ambiente fechado jpeado anteriormente, retornando a posi do usuo.

3.2.4 Ponto de Vista da Engenharia

Para o uso do sistema sercesso que o dispositivo m que executar o aplicativo tenha uma antena de *Wi-fi*, para que possam ser realizadas as medis de intensidades dos sinais. Tais dados sernviados pela pra antena de *Wi-fi* usando a rede que estiver conectada ou pela antena de 3G/4G usando as redes ms para o servidor. O serviuncionar uma mina virtual hospedada no servimazon AWS, sendo que esta mina deverr capaz de execu do algoritmo de *machine learning*.

ecesso tambue exista uma estrutura mma no local a ser mapeado, deve existir um nmero mmo de pontos de acesso de rede sem fio suficiente para englobar todo o ambiente. Esta

cobertura neve ser apenas de pelo menos um sinal *Wi-fi* na parcela de a, dispensl que mais de um sinal atinja cada ponto para que seja possl haver a determina do ponto escolhido.

3.2.5 Ponto de Vista da Tecnologia

O processamento e aplica dos algoritmos de *machine learning* sereitos por meio da linguagem R. O aplicativo ito para a plataforma Android e o servidor plementado na plataforma *Ruby on Rails*, hosteado em servidor Amazon AWS, para pequenas aplicas que devem rodar na nuvem. O R rvido por meio de uma plataforma REST por meio da biblioteca Plumber, que permite que chamadas as funs do R sejam feitas por meio de requisis HTTP.

4 METODOLOGIA

O projeto desde a sua idealiza foi gerenciado com metodologia I, com o grupo sempre pensando em *sprints* com pequenos protos entregis. A ferramenta usada pelo grupo foi o **Trello**. Foi sempre feita a distin entre pesquisa, documenta e implementa. Como pico de metodologias is, os testes sempre foram realizados em paralelo com a implementa de cada nova *feature*, de modo que para cada nova parte do projeto implementada, possuos testes para garantir que as funs antigas ainda funcionam individualmente.

5 MACHINE LEARNING

5.1 Tratamento dos dados

5.1.1 Dados obtidos do Reposit UCI

Para grande parte dos testes de *cross-validation*, foi usado o *dataset UJIIndoorLoc* apresentado em [**?**]. O *dataset* consiste em diversas medidas de potia de sinal *Wi-fi* medidas com diversos aparelhos celulares em 3 pros diferentes de uma faculdade. As medidas esteparadas por pro, andar e sala. Para os fins dessa monografia, serealizados testes com medidas sempre de um mesmo andar, afim de classificar as zonas de um mesmo ambiente. Esse *dataset* usa a constante 100 para designar valores nedidos. Para come o tratamento os substitus pelo valor de -120 (dB), que para todos os fins prcos, a medida de potia que representa um valor nulo. E finalmente, antes de serem usados para treinar os modelos, os dados sransformados de modo a ficarem com ma 0 e variia 1 por coluna, o que cesso para evitar comportamentos indesejados dos algoritmos de ML.

A transforma alizada da seguinte maneira, com X_j representando a j-ma coluna da matriz de dados, com ma μ e desvio-padrσ, para cada elemento i dessa coluna:

$$X_{ij} = \frac{X_{ij} - \mu}{\sigma}$$

5.1.2 Dados do Servidor

De modo ango aos dados pegos do Reposit UCI, a matriz de dados ansformada para ter ma nula e variia unita em cada coluna, sendo que antes disso valores nulos substitus por -120.

5.2 Formato dos dados tratados

A matriz de dados ,depois de tratada, tem o seguinte formato:

ccccccc $ZoneID$ $BSSID_1$ $BSSID_2$ ... $BSSID_n$
(ccccc)c 1-70 -92 ... -87 $Measure_1$
2-89 -80 ... -63 $Measure_2$
3-28 -120 ... -35 $Measure_3$
$\vdots\vdots\vdots\ddots\vdots\vdots$
1-48 -36 ... -29 $Measure_n$

Cada coluna representa a medida de uma *BSSID* (i.e. um Roteador), ou seja, uma *feature* para o ML, e cada linha ponto de treino (i.e. uma medida para cada *BSSID*, nossas *features*).

5.3 Modelos Usados e Cross-Validation de partros

A chamada *cross-validation* dos modelos de ML este para encontrar os partros os para o treinamento. Um dos mdos escolhidos de *cross-validation* foi o *k-fold*. O mdo *k-fold* divide o dataset em K subconjuntos de igual tamanho e entm dos conjuntos ado como valida do treinamento feito pelos $K-1$ subconjuntos restantes. O processo petido K vezes e em cada subdivisossl podem ser usados valores de partros de treino distintos. Para a compara dos modelos, sereitos diversos testes com um nmero crescente de zonas de classifica. Nesse caso, sscolhidas iterativamente e aleatoriamente n zonas do *dataset UJIIndoorLoc*, segos todos os pontos associados as n zonas selecionadas. Ent o *dataset* parado em 80% de pontos de treino e 20% para testes. Para valores de n de 1 a 30 sinalmente medidos as taxas de erro de cada algoritmo e essa informa resentada em grcos.

5.3.1 KNN : K-Nearest-Neighbors

Para o algoritmo de KNN, foram feitos dois testes de *cross-validation*. No primeiro, a mica de distia foi decidida, e no segundo, o nmero de vizinhos (K). Para que fosse definida a mica de distia, foi usado o mdo *k-fold* com $K=2$. E em cada *fold* o modelo foi treinado com uma mica diferente. Foram calculados 20 conjuntos diferentes de *folds* e foi tirada a ma do erro de valida para cada uma das micas.

Depois, foi usado o mdo *k-fold* com $K=10$ para a deciso nmero de vizinhos. Foram testados os valores de 1 at para o nmero de vizinhos. Na imagem a seguir vemos detalhes desse teste.

Podemos concluir entue (1) o erro nor no geral com a distia de *Manhattan* e (2) embora

tenha muita variia envolvida no teste, o valor de K que minimiza o erro fica proximo de 4 vizinhos.

No restante do trabalho foi escolhido o valor de $K = 4$ vizinhos.

5.3.2 Rede Neural

Para a *cross-validation* das Redes Neurais, tamboi usado o mdo de *k-fold* com $K = 10$. Para cada uns dos*folds* foi testado um nmero de neurnios em uma *hidden-layer* nica. (Outros testes mostraram naler a pena para o nosso problema usar mais de uma camada de *hidden layer* ou *deep learning*). Os resultados sostrados a seguir:

Conclus que o nmero ideal de neurnios na *hidden-layer* esto de 200. E se valor que usaremos daqui para frente.

5.3.3 Arvore de decis

Baseado no trabalho apresentado por [**?**], foi escolhida uma implementa do algoritmo C4.5 [**?**] em Java, da biblioteca Weka, para serem geradas ores de decis Os mdos pororam chamados pela interface da Weka para R, chamada RWeka. A fun J48 do Weka foi testada com o os pontos de treino referentes ao primeiro andar do pro 1 (i.e. BuildingID 1, FloorID 0) do dataset *UJIIndoorLoc*. Foi levantada a curva de erro de classifica para uma quantidade crescente de pontos de treino (como explicado em 2.3).

5.3.3.1 Acoplamento com o Algoritmo AdaBoost

Boosting deia em ML de criar uma predi forte e precisa combinando diversas previses mais fracas. O algoritmo AdaBoost [**?**] mdo para melhorar a precise classificadores fracos (i.e. *Weak Learners*) por meio de uma vota ponderada que leva em conta o erro de diversos votadores fracos treinados no processo. rovado que o modelo final se torna um classificador forte [**?**]. Por defini, um classificador fraco uele que consistentemente consegue ser melhor que um chute para a classifica (e.g. o erro nor que 50% para o caso de duas classes possis de classifica).

Tambaseados em [**?**] e [**?**]. Usaremos o algoritmo AdaBoost e sua implementa na biblioteca Weka para melhorar a precisas ores de decis4.5. A seguir vemos um grco comparando a classifica para o mesmo andar do *dataset UJIIndoorLoc* com os mesmos pontos de teste e uma quantidade crescente de pontos de treino, assim como feito no to anterior.

Podemos notar que com um nmero grande de pontos de treino, obtemos uma melhora sensl de precisara a classifica, e portanto, no restante do trabalho, iremos usar o algoritmo C4.5 acoplado com o algoritmo AdaBoost.

5.3.4 Compara dos Modelos Usados

Reiterando o que foi explicado em outra sess os testes a seguir irontemplar todos os modelos usados. Os modelos tem seu erro testado para datasets com um nmero crescente de zonas aleats a serem classificadas. Todos os testes foram feitos com os dados do primeiro andar do pro 1 do dataset *UJIIndoorLoc* (i.e. BuildingID 1, FloorID 0).

Figura 5: Erros para uso de Diferentes Distias

Figura 6: Erro em fun do nmero de vizinhos

Figura 7: Erro para diversas quantidades de neurnios na rede neural.

Figura 8: Teste de Valida do Algoritmo C4.5

Figura 9: Compara do C4.5 com o uso do AdaBoost

Figura 10: Erro do algoritmo SMO para uma quantidade crescente de zonas de classifica.

Figura 11: Erro do algoritmo de Redes Neurais para uma quantidade crescente de zonas de classifica.
Figura 11: Erro do algoritmo de Redes Neurais para uma quantidade crescente de zonas de classifica.

Figura 12: Erro do algoritmo KNN para uma quantidade crescente de zonas de classifica.

Figura 13: Erro do algoritmo de rvore de Decisom e sem o Adaboost para uma quantidade crescente de zonas de classifica.
Figura 13: Erro do algoritmo de rvore de Decisom e sem o Adaboost para uma quantidade crescente de

5.4 Mdos de Vota

Os chamados sistemas de *Ensemble Learning* squeles em que uma classifica ita com base em diversos modelos treinados. Em tos anteriores foi discutido o uso do algoritmo AdaBoost, que mdo desse tipo para melhorar o poder de classificadores fracos treinando iterativamente diversos modelos fracos progressivamente com os erros do anterior [**?**]. Por para agregar os modelos usados atora, iremos implementar algo mais simples, poreguindo a mesma la: Sistemas de Vota. Baseados em [**?**], usaremos diversos mdos para combinar as classificas dos nossos modelos treinados. Em um, usaremos apenas a classe de sa dos modelos, e posteriormente, as probabilidades (*supports*) para cada uma das classes, que tambas dos modelos. Esses *supports* podem ser, dependendo do modelo, a probabilidade posterior ou o grau de confianornecido pelos algoritmos. Por exemplo, nas redes neurais a sa de um neurnio un sigmoide aplicada nas entradas multiplicada pela matriz de pesos da rede, que, por defini, nmero entre 0 e 1 que representa uma probabilidade.

5.4.1 Testes com os Mdos de Vota

Como na sesse compara dos modelos, os mdos tem seu erro testado para datasets com um nmero crescente de zonas aleats a serem classificadas. Todos os testes foram feitos com os dados do primeiro andar do pro 1 do dataset *UJIIndoorLoc* (i.e. BuildingID 1, FloorID 0).

Figura 14: Erro do Voto Simples e Voto Ponderado para uma quantidade crescente de zonas de classifica.

Figura 14: Erro do Voto Simples e Voto Ponderado para uma quantidade crescente de zonas de classifica