Localização indoor através de sinal WLAN

Gustavo Henrique do Nascimento Pereira
gunasper@gmail.com

October 17, 2015

1 Introdução

Nos últimos anos, métodos para sistemas de localização em ambientes fechados usando redes WiFi vem sido sugeridos. Esse tipo de serviço tornou-se necessário uma vez que permitem que diversas atividades computacionais (entrega de conteúdo, rastreamento, etc) possam ser feitas com maior precisão.

Um dos principais métodos consiste na utilização de Access Points (AP, sensores) para sensoriar o espectro WiFi e coletar o sinal WiFi que dispositivos móveis enviam periodicamente. Assim, para cada sinal enviado por um dispositivo móvel, é coletado o received signal strength (RSS), um valor negativo medido em dBm que pode ser visto como a força em que o rádio de um AP sente o sinal enviado por dispositivos móveis em seu raio de alcance. A partir disso, encontra-se o valor médio dos sinais recebidos naquela área, e esse valor passa a ser o valor esperado para o sinal que chega num dispositivo.

A técnica mencionada anteriormente é conhecida como técnica de avaliaçaõ de fingerprint, e é relativamente simples comparada a outras técnicas, como Angle-of-Arrival (AOA) ou Time-of-Arrival (TOA). Além disso, nenhum hardware específico é necessário no dispositivo móvel, e qualquer infraestrutura wireless pode ser reaproveitada nesse sistema.

Assim, este trabalho tem como objetivo um primeiro estudo sobre sistemas de localização interna, avaliando a técnica de fingerprint na base de dados oferecida, bem como sugerir algoritmos alternativos para resolução desse tipo de problema.

2 Dataset

A base de dados fornecida consiste em tuplas do tipo (Área, Dispositivo, [APs]). Área é o codenome dado a um ambiente físico que determina aonde um dispositivo mobile está localizado. O dispositivo é um identificador que serve para dizer sobre qual dispositivo estamos tratando. [APs] é um vetor contendo os received signal strength (RSS) fornecidos por cada AP sobre o dispositivo.

A base é dividida em dois arquivos: treino e teste. O primeiro deve ser usado para treinar um algoritmo capaz de aprender os padrões dos sinais RSS de uma determinada área. O segundo arquivo possui sinais que devem ser usados para verificar se o algoritmo usado para treino e classificação foi bem sucedido. Estes sinais possuem um campo com a área em que estão, e deve-se apenas checar se a área apontada pelo algoritmo corresponde a área definida para o sinal.

A base de treino possui exatas 5501 amostras das 34 áreas possíveis. A figura 1 mostra como essas amostras estão distribuídas entre as várias áreas possíveis. Nota-se que as áreas A001, A033Parking e A033Front são as áreas com maior número de amostras disponíveis.

O fato de termos muitas amostras para cada área é útil uma vez que os valores de RSS podem variar bastante dentro da mesma área como mostra a figura 2. Um grande número de amostras permite que o RSS médio de uma área seja calculado de forma mais precisa, evitando que valores muito grandes ou muito pequenos distorçam a média. Entretanto, para algumas classes de algoritmos que podem ser usados para resolver este problema, esse desbalanceamento na base pode vir a trazer problemas, como veremos mais a frente.

A alta variação nos valores de RSS medidos numa mesma área é mostrada na figura 2. As barras vermelhas indicam a média dos valores de RSS encontrados na base de dados. As barras pretas (acima das vermelhas) indicam o valor mínimo e máximo do RSS num AP. Além disso, foi calculada a variância desses valores. Para área A001, a variância variou entre 9 e 11 pontos para cada AP. A alta variância pode comprometer os resultados obtidos, uma vez que diversos vetores de RSS podem acabar pertencendo a uma mesma área.

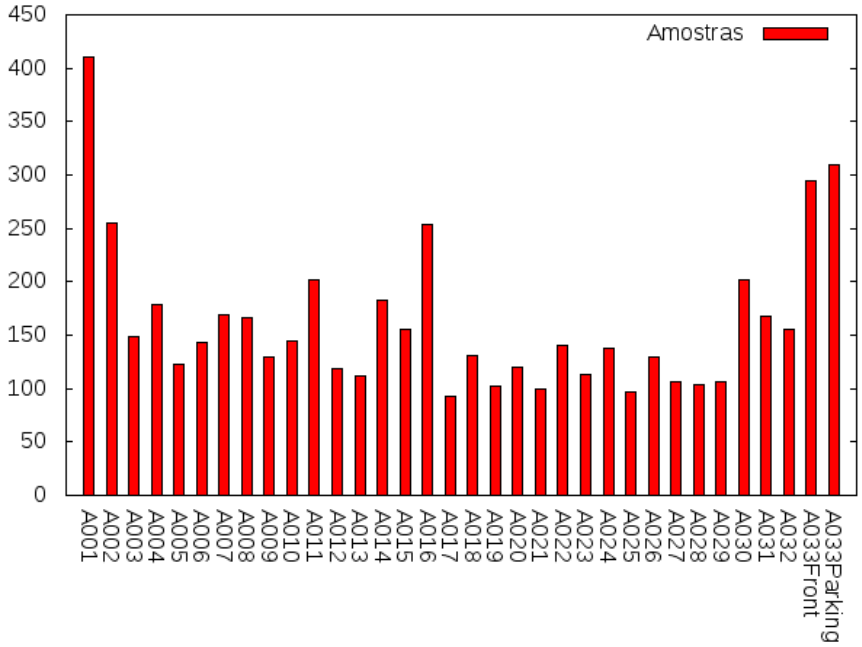

Figure 1: Base de treino: número de amostras disponíveis agrupadas por área

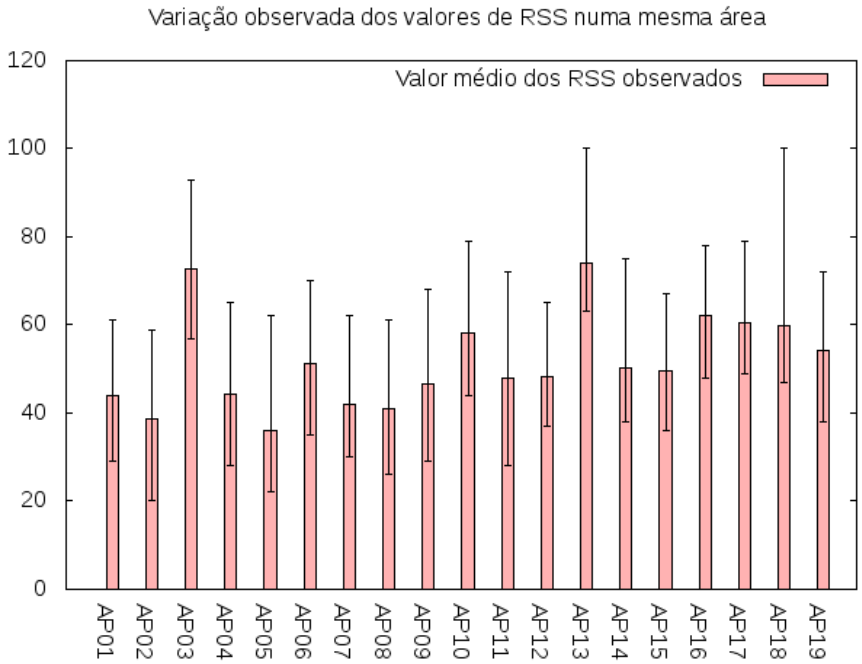

Figure 2: Base de treino: variação dos RSS nos 19 AP na área A001

3 Soluções Propostas

Para resolver o problema de localização indoor usando WLAN, as soluções encontradas possuem duas fases: treino (também chamada de fase offline) e testes (fase online). Na fase de treino, os algoritmos são treinados utilizando uma base de dados com fingerprints e localizações já conhecidas. Na fase de testes, os algoritmos tentam prever aonde um dispositivo está baseando-se no fingerprint do dispositivo naquele momento e fazendo comparações com os fingerprints obtidos na primeira fase.

3.1 Algoritmos baseados na avaliação de fingerprints

A primeira solução implementada foi modelada e estudada em [3]. Para estimar a localização de um ponto P, o algoritmo computa a a distância euclidiana entre os valores de RSS medidos naquele ponto (que forma um fingerprint daquele ponto, naquele momento), com os os valores armazenados na base de dados. A localização de P é então estimada como sendo a localização do fingerprint da base de dados que possuir menor distância em relação a P.

Na fase de treino, o algoritmo proposto computa a média dos valores de RSS observados numa determinada área. Essa média torna-se, então, o fingerprint daquela área. O procedimento acontece até que obtenha-se o fingerprint de todas as áreas. Logo em seguida, durante a fase de testes, a comparação é feita com os valores médios armazenados. Entretanto, como apresentado em [4], os valores de RSS para uma dada área podem sofrer alteração devido a diversas circunstâncias. Para contornar esse problema, os autores sugerem que sejam utilizados diversos fingerprints para a mesma área, e que a localização, então, seja estimada baseando-se na distância a esses fingerprints. Ambas as versões foram implementadas e testadas. A primeira versão nos oferece um baseline mínimo de precisão.

Para este trabalho, foram considerados 19 APs. Apesar de não ser um número muito grande (> 100), essa quantidade de dimensões pode ser suficiente para comprometer o cálculo da distância, devido a maldição da dimensionalidade[1]. Para contornar esse problema, substitui a distância euclidiana pela distância de cosseno[2], uma métrica mais confiável para distância em bases multidimensionais. Também foram construídas versões utilizando a distância de cosseno que faz uso de vetores médios e também de todos os fingerprints da base de dados.

3.2 Algoritmos baseados em Mineração de Dados

A Mineração de Dados é geralmente dividida em 3 subáreas: Regras de Associação, Agrupamento e Classificação. Algoritmos de classificação (também chamados de aprendizado supervisionado) inferem/descobrem a classe de um objeto o a partir de outros objetos de classe já conhecida. Para tanto, esses algoritmos também se dividem em duas fases: treino e testes. Similares aos algoritmos de avaliação de fingerprints, a fase de treino é feita utilizando a base de dados conhecida e, geralmente, são "aprendidas" regras sobre a base. Essas regras são usadas na fase de testes para que, então, possa prever a qual classe o objeto o pertence.

No contexto deste trabalho, um objeto é um dispositivo móvel numa determinada área. As classes são as áreas nas quais esse dispositivo pode estar. Para encontrá-las, são avaliadas os atributos desse dispositivo. Os atributos, naturalmente, são os sinais RSS medidos.

Os dois algoritmos de classificação que obtiveram mais acertos foram o K-Nearest-Neighbors (KNN) e o Support Vector Classifier (SVC). Ambos os algoritmos tratam a base de dados como um espaço de 19 dimensões. Cada fingerprint é colocado nesse espaço durante a fase de treino.

A diferença entre eles está no processamento feio a seguir. No KNN, ao tentar classificar um novo objeto o, este é colocado na base de dados e então busca-se os K vizinhos mais próximos dele (baseado numa métrica de distância euclidiana - minkowski, com p = 2). Os K vizinhos mais próximos são então testados quanto a classe que pertencem. A classe com maior número de exemplares entre os K objetos é escolhida como a classe de o.

No SVC, a abordagem é diferente. O algoritmo, através de formulações matemáticas, encontra vetores suporte que delimitam as fronteiras entre uma classe e o outra, de forma que a base de treino possua o menor erro possível. É possível, ainda, utilizar um kernel polinomial, que faz com que esses vetores tornem-se polinômios, a fim de obter maior precisão. Entretanto, neste trabalho, o uso do kernel não trouxe ganhos reais, e o tempo de execução excedeu em várias vezes a versão com vetores lineares.

[1] https://en.wikipedia.org/wiki/Curse_of_dimensionality
[2] https://en.wikipedia.org/wiki/Cosine_similarity

4 Análise de Resultados

4.1 Fingerprint utilizando um vetor de valores médios

Durante os primeiros experimentos, a técnica descrita em [3] foi utilizada. O vetor de RSS médio para cada área foi inicialmente calculado, tornando-se, assim, o fingerprint daquela área e salvo na base de treino. Em seguida, para cada vetor de RSS da base de testes V, foi calculada a distância entre V e os n fingerprints F_n da base de treino. V, então, recebia o valor de F_n cuja distância $V - F_n$ fosse a menor entre todas as distâncias calculadas.

Uma vez que [3] afirma que para ter uma boa precisão deve-se ter um devio padrão entre 2 e 4, e o desvio padraõ da base de treino é, em geral, entre 9 e 11, foi feito um teste usando a mediana (e não a média) para gerar o fingerprint. Como falado anteriormente, a distância euclidiana nem sempre é a melhor métrica de distância, principalmente em bases com muitas dimensões. Foi feito, assim, um comparativo entre os acertos utilizando a distância euclidiana e a distância de cosseno. Como podemos ver, a distância de cosseno se saiu melhor em todos os casos comparados.

	euclidiana	cosseno	métricas
média do fingerprint	740	1456	#acertos
	20%	41%	precisao
todas as amostras	1120	1039	#acertos
	31%	29%	precisao
mediana do fingerprint	868	1337	#acertos
	24%	37%	precisao

Table 1: Resultados aplicando a metodologia de fingerprint para as 3543 entradas da base de testes

4.2 K-Nearest-Neighbors

O KNN foi aplicado utilizando vários valores de K. Os melhores resultados obtidos são mostrados na tabela 2. Por ela, podemos observar que o KNN teve uma média de acertos similar ao algoritmo usando a média dos fingerprints, apesar de levemente inferior. O parâmetro K além de 70 abaixou a precisão, o que fez com que não seja incluído na tabela.

valor de K	#acertos	precisão
10	1337	37 %
30	1401	39 %
50	1390	39 %
70	1352	38 %

Table 2: Resultados aplicando o K-Nearest Neighbors

4.3 Support Vector Classifier

De todas as abordagens testadas, o SVC foi o que possuiu o melhor resultado em termos de quantidade de acertos. Para ambos os kernels foram aplicados três parâmetros de penalidade (1, 2 e 5), porém, isso não levou a uma diferença significativa de resultados e assim esses valores forem omitidos. Na tabela 3 consta apenas os valores obtidos para penalidade 1. O kernel polinomial obteve uma precisão levemente menor que o kernel linear, a um custo computacional muito maior, o que faz com que seu uso não seja recomendado.

valor de K	#acertos	precisão
linear	1525	43%
polinomial	1361	38%

Table 3: Resultados aplicando o Support Vector Classifier

4.3.1 Outros algoritmos

Foram testados, ainda, outros algoritmos clássicos de classificação: nearest centroid, decision tree, mas nenhum deles obteve um desempenho satisfatório o bastante para ser incluso nesse relatório. Devido ao prazo dado para o trabalho, não foram investigadas as razões para a baixa qualidade da classificação oferecida.

4.4 Análise por área do KNN e do SVC

Apesar de os valores de precisão terem ficado bastante similares ao método de fingerprints usando distância de cosseno, a precisão de 43% (a melhor alcançada) não é boa para um sistema de localização. Devido a isso, foram investigadas possíveis razões para um resoltado tão ruim. Algumas teorias desenvolvidas são explicadas logo abaixo.

4.4.1 Alto desvio padrão

A base de treino (bem como a de testes) possui um desvio padrão muito alto. Alguns valores de RSS, como mostrado na primeira sessão, possuem uma variação de até 50 pontos. Uma vez que o RSS varia de -1 até -100, isso significaria que a intensidade do sinal RSS num AP pode assumir a metade dos valores possíveis na escala, o que torna a localização bastante ineficaz, mesmo com um número grande de AP. O motivo de um alto desvio padrão deve-se tanto as variações e oscilações das ondas WLAN quanto ao tamanho da área em que elas foram medidas. É natural imaginar que as maiores áreas como A033.Parking e A033.Front possuam as maiores variações nos sinais de RSS. Para corrigir esse problema, [3] sugere que a área de medições seja de cerca de 1,25 metro^2, para prevenir altas oscilações nos vetores de RSS.

A posição que um dispositivo se encontra também pode oferecer uma variação significativa no valor do RSS. [4] sugere que as medições sejam feitas em diversos ângulos por área e criar mais de um único fingerprint para cada área.

4.4.2 Base de treino desbalanceada

A base de dados de treino encontra-se desbalanceada quando uma determinada classe possui mais representantes do que as demais. Como podemos ver pela figura 1, a calsse A001 possui quase 5 vezes mais representantes que a classe A017. Por um lado isso é bom, pois temos um valor mais confiável das dimensões dos vetores RSS em A001. Por outro lado, isso torna o classificador KNN bastante enviesado, uma vez que a grande quantidade pontos de A001 pode ser vizinha de quaisquer outros pontos. De fato, isso faz com que o número de tentativas em A001 tenha sido muito maior que o número de tentativas em A017 quando o KNN foi aplicado.

Para área A001, 92 é o número de vetores que foram marcados como A001 e que, de fato, são A001. Os falsos negativos são vetores que são A001 mas que foram marcados em outra categoria. A precisão é o número de acertos da área A001 sobre o número de previsões em A001. Por fim, a revocação é calculada como sendo o número de A001 encontrados dividido pelo número de A001 existentes na base.

Como podemos ver em 4, o fato de termos muitas amostras em A001 fez com que a revocação tenha sido alta (pudemos recuperar mais elementos que estejam de fato em A001), porém, a precisão caiu bastante, uma vez que outros 164 elementos foram marcados como A001 quando na verdade estavam em outras áreas.

Para resolver esse tipo de problema, idealmente trabalha-se com uma base de treinos balanceada. Caso não esteja, pode-se, ainda, utilizar de estratégias de Undersampling, Oversampling ou Cost-Sensitive Training, como mostrado em [1].

4.4.3 Uso de métodos que levem em conta a posição anterior do dispositivo

Em [2] diversas melhorias são propostas a um dispositivo de localização chamado RADAR. Entre essas melhorias, é sugerido considerar o histórico de localização do dispositivo a ser rastreado. A melhoria deve-se ao fenômedo chamado Aliasing, que acontece quando dois pontos, apesar de fisicamente distantes, possuem fingerprints similares. Assim, durante a fase de classificação, se um o sinal analisado estiver apontando para longe de sua localização anterior para um determinado dispositivo, esta localização será descartada e não será considerada para a inferência da posição atual.

Area	Acertos	Falso Negativo	Tentativas	Precisão	Revocação
A001	93	15	261	0.36	0.86
A002	52	56	110	0.47	0.48
A003	30	64	47	0.64	0.32
A004	54	50	103	0.52	0.52
A005	11	93	22	0.50	0.11
A006	17	71	73	0.23	0.19
A007	51	56	124	0.41	0.48
A008	38	50	90	0.42	0.43
A009	23	80	52	0.44	0.22
A010	43	65	113	0.38	0.40
A011	58	34	128	0.45	0.63
A012	42	43	95	0.44	0.49
A013	11	108	29	0.38	0.09
A014	45	45	164	0.27	0.50
A015	50	58	123	0.41	0.46
A016	58	43	198	0.29	0.57
A017	28	79	52	0.54	0.26
A018	32	77	90	0.36	0.29
A019	29	78	65	0.45	0.27
A020	39	70	74	0.53	0.36
A021	14	81	42	0.33	0.15
A022	44	70	114	0.39	0.39
A023	30	76	69	0.43	0.28
A024	37	76	113	0.33	0.33
A025	13	106	30	0.43	0.11
A026	70	46	168	0.42	0.60
A027	28	66	71	0.39	0.30
A028	24	93	68	0.35	0.21
A029	45	68	104	0.43	0.40
A030	56	29	181	0.31	0.66
A031	19	90	64	0.30	0.17
A032	61	46	181	0.34	0.57
A033Front	67	46	132	0.51	0.59
A033Parking	89	14	193	0.46	0.86

Table 4: Resultados por área obtidos ao aplicar o KNN

5 Conclusão

Neste trabalho foi possível conhecer sobre localização indoor usando fingerprints. Foi possível, ainda, verificar diversos problemas de coleta de dados que podem oferecer complicações diversas durante a execução do algoritmo de localização. Vários algoritmos foram testados e foi comprovado, na prática, que a técnica de fingerprint é tão boa quando algoritmos de aprendizado de máquina. Apesar do SVM ter alcançado uma precisão superior, este é mais caro computacionalmente e deve ser feito um estudo mais amplo para saber se há a necessidade e se é possível usá-lo para a aplicação desejada.

References

[1] Class imbalance in supervised machine learning. http://stats.stackexchange.com/questions/131255/class-imbalance-in-supervised-machine-learning. Accessed: 2015-10-15.

[2] Paramvir Bahl, Venkata N Padmanabhan, and Anand Balachandran. Enhancements to the radar user location and tracking system. Technical report, technical report, Microsoft Research, 2000.

[3] Kamol Kaemarungsi and Prashant Krishnamurthy. Modeling of indoor positioning systems based on location fingerprinting. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 1012–1022. IEEE, 2004.

[4] Apostolia Papapostolou and Hakima Chaouchi. Wife: wireless indoor positioning based on fingerprint evaluation. In *NETWORKING 2009*, pages 234–247. Springer, 2009.