Bewegingsherkenning met een smartphone

Arne De Brabandere
arne.debrabandere@student.kuleuven.be

Menno Keustermans
menno.keustermans@student.kuleuven.be

Abstract

1 Inleiding

2 Afzondelijke activiteiten

Het eerste probleem is om van een gegeven reeks samples van de accelerometer en gyroscoop van een smartphone de activiteit van een persoon te bepalen. We veronderstellen hier dat telkens één afzonderlijke activiteit gemeten wordt.

We willen tien verschillende activiteiten kunnen herkennen:

- wandelen,
- lopen,
- fietsen,
- een trap opwandelen,
- een trap afwandelen,
- springen,
- niets doen (zitten, liggen, staan),
- een lift naar boven nemen,
- een lift naar beneden nemen,
- tanden poetsen.

In bovenstaande lijst hebben we tanden poetsen als moeilijke activiteit toegevoegd. De beweging lijkt sterk op niets doen en zal waarschijnlijk minder goed te herkennen zijn. Ook voor lift naar boven en trap opwandelen bestaan er gelijkaardige bewegingen, respectievelijk lift naar beneden en trap afwandelen.

Het proces om een afzonderlijke activiteiten te herkennen verloopt in drie stappen. Ten eerste moeten er gegevens verzameld worden. Als tweede stap worden er features berekend op de verzamelde gegevens. Deze zijn nodig om tenslotte modellen te leren met behulp van classificatiemethodes. Als criterium om de verschillende modellen te vergelijken, gebruiken we de accuraatheid als percentage van het aantal juiste geclassificeerde samples ten op zichte van het totaal aantal samples met behulp van cross-validatie.

2.1 Gegevensverzameling

Alle gegevens werd opgemeten door MotionTracker tool . Dit is een Android-applicatie die de versnelling en rotatie (respectievelijk gemeten door de accelerometer en gyroscoop van de smartphone) aan 50 Hertz. Als uitvoer geeft de applicatie een .log-bestand met de gemeten versnellingen (in de x-as, y-as en z-as met de z-as evenwijdig met de gravitatie) en rotaties (in quaternion notatie) met bijhorende timestamps.

Voor elke meting werd de applicatie gestart alvorens de smartphone in de broekzak gestopt werd en gestopt na het uithalen. Waardoor er bij het begin en einde van elke meting altijd enkele seconden niet-activiteit bevat. Om zo het fout labelen te vermijden, werd na elke meting een stuk van de start en het einde van het .log-bestand weggeknipt. Zodat elke meting exact één activiteit bevat van vier á twintig seconden.

We hebben voor elke activiteit 22 metingen verzameld, opgemeten door twee verschillende personen. Om voldoende variatie te hebben, gebeurden de metingen op verschillende dagen. Ook hebben we ervoor gezorgd dat we niet telkens dezelfde broek droegen, aangezien de gemeten versnelling kan variëren in verschillende broekzakken. Na elke meting werd het uitgevoerde .log-bestand geknipt en gelabeld met de juiste activiteit.

2.2 Gegevensverwerking: features berekenen

Voor we classificatiemethodes kunnen gebruiken, moeten we eerst features berekenen. Dit zijn parameters die we uit de samples van de accelerometer en gyroscoop kunnen halen. Om de verschillende features te berekenen, maakten we gebruiken van MotionFingerPrint.jar tool

MotionFingerPrint berekent in totaal 134 features verdeeld onder vier soorten:

- *Statistische features:* dit zijn gemiddelde, standaardafwijking van zowel z- als xy-versnellingen en vermogen. En correlatie tussen z- en xy-versnelling.
- *Fourier-transformatie:* deze worden berekend in het frequentie domein van de metingen, zoals amplitudes volgens de verschillende assen.
- *Wavelet-transformatie:*
- *Hidden Markov models:* log-likelihood voor het model van elke activiteit

Het resultaat van de features berekening geeft een voor elke sample een set van parameters. Elke set vormt een instantie van de training set om een model te leren.

Feature selectie

Tussen de verschillende soorten features is er verschil in berekeningstijd en hoeveelheid informatie. Zo kunnen statistische features in het algemeen tegen een lagere kost berekend worden ten op zichte van fourier transformatie features. In sommige toepassingen kan het daarom interessant zijn om slechts een deel van het totaal aantal features te berekenen en juist degene die het meeste informatie bevatten.

Met behulp van feature selectie hebben we onderzocht hoeveel features van het totaal aantal features van elke soort nodig zijn om een redelijke accuraatheid te behalen. Om de features met het meeste informatie te vinden, maakten we gebruik van weka's: *InfoGainAttributeEval* klasse. Deze klasse evalueert de waarde van elke features met behulp van information gain. Hierbij werd er steeds de entropie waarde berekend.

Vervolgens om de geselecteerde features te evalueren maakten we gebruik van dubbele cross-validatie. Bij de eerste cross-validatie (10-fold) werd de gegevensset verdeeld in een training- en testset. Daarna gebruikten we de gegeneerde training-set om de beste features te selecteren met het proces dat hierboven werd beschreven gebruik makend van de tweede cross-validatie (2-fold). Met de geselecteerde features werd een model opgesteld met behulp van de randomForest classificatiemethode. En dit model werd uiteindelijk getest om de testset die bij de eerste cross-validatie gemaakt werd.

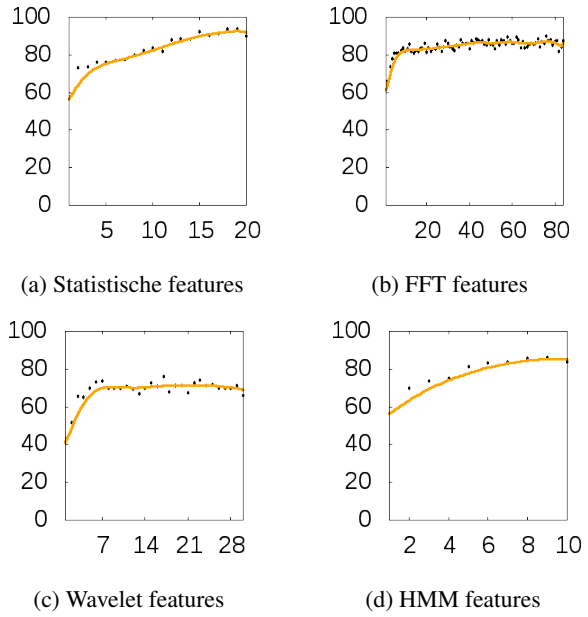
In figuur X worden de resulaten voor de verschillende soorten features getoond. We zien dat we met enkel de statistische features al een accuraatheid van 90% behalen. Zelfs met de helft kan het model al vrij zeker de juiste activiteit voorspellen. We stellen vast dat van het grote aantal fourier tranformatie features slechts een tiental nodig zijn om een nauwkeurigheid van 80% te bekomen. In het algemeen kunnen we kunnen concluderen dat wavelet transformatie functie minder goed werken (slechts 70% accuraatheid). Tenslotte in de laatste grafiek merken we op dat de Hidden Markov Models enkel degelijk werken als alle features gebruikt worden.

Voor de rest van het onderzoek werd gebruik gemaakt van de volledige set features.

2.3 Classificatiemethodes

We gebruiken classificatiemethodes om een model te zoeken om afzonderlijke activiteiten te herkennen. We vergelijken enkele veel voorkomende methodes: beslissingsbomen, Random Forest, k-Nearest Neighbours, Naive Bayes en Support Vector Machines. De methodes leren telkens een model uit de instanties met de hierboven beschreven features voor de verschillende activiteiten. Hieronder wordt kort de werking van de vernoemde methodes beschreven:

- *Beslissingsbomen* zijn bomen waarvan de interne knopen features voorstellen en de bladknopen één (of meerdere) labels bevatten. Een tak in de boom stelt een test voor op de feature van de knoop waaruit de tak vertrekt. Om een nieuwe instantie te classificeren worden de juiste takken gevolgd tot in een bepaalde bladknoop. De instantie wordt dan geclassificeerd als het label in die bladknoop.



(a) Statistische features (b) FFT features

(c) Wavelet features (d) HMM features

Figuur 1: Resultaten van het feature selectie experiment. Op de x-as wordt steeds het aantal features van de soort features geplot en op de y-as de accuraatheid in procenten.

- *Random Forests* zijn combinaties van beslissingsbomen met een willekeurig deel van de trainingset en een willekeurig deel van de features. Bij de classificatie van een nieuwe instantie 'stemt' elke boom op een label. De instantie krijgt vervolgens het label met de meeste stemmen.
- *k-Nearest Neighbours* classificeert nieuwe instanties door te zoeken naar de k dichtstbijzijnde instanties in de trainingset. Dit zijn de instanties waarvan de waarden voor de features het dichtst bij die van de te classificeren instantie ligt. Hieruit wordt het meest voorkomende label gekozen.
- *Naive Bayes* is een probabilistische classifier die gebruik maakt van voorwaardelijke kansen. De kans op een label C met gegeven waarden voor de features $\mathbf{F} = (F_1, ..., F_n)$ wordt berekend als $P(C|\mathbf{F}) = \frac{P(\mathbf{F}|C)P(C)}{P(\mathbf{F})}$ waarin $P(\mathbf{F}|C)$, $P(C)$ en $P(\mathbf{F})$ uit de trainingset kunnen berekend worden. Bij het classificeren wordt het label met de grootste kans gekozen.
- *Support Vector Machines* beschouwt de features als een multidimensionele ruimte. De instanties zijn dan punten in de featureruimte. Bij het leren worden de instanties lineair van elkaar gescheiden door hypervlakken, zodat de instanties aan de ene kant van het vlak een ander label hebben dan die aan de andere kant. Wanneer de instanties niet lineair te scheiden zijn, worden ze getransformeerd met een *kernel functie*. We gebruiken hier LibSVM, waarbij binaire (één-tegen-één) classifica-

tie gedaan wordt: elk hypervlak scheidt de instanties van 2 verschillende labels. Een nieuwe instantie wordt geclassificeerd door elke binaire classificatie als een stem voor een label te beschouwen. Het label met het grootste aantal stemmen wordt gekozen.

2.4 Experimenten en resultaten

We evalueren de methodes met 10-fold cross-validatie. Hierbij worden de instanties op 10 verschillende manieren opgesplitst zodat telkens 90% van de instanties als trainingset wordt gebruikt om een model uit te leren. Op de overige 10% wordt het model geëvalueerd. De accuraatheid van elke methode wordt dan berekend als het gemiddelde percentage van correct geclassificeerde instanties in de verschillende testsets.

In figuur X wordt de accuraatheid van de verschillende methodes vergeleken. We zien dat Random Forest de hoogste accuraatheid geeft voor onze metingen. Ook beslissingsbomen (J48) geeft een goede accuraatheid, net zoals Naive Bayes. De hogere accuraatheid van Random Forest ten opzichte van beslissingsbomen is waarschijnlijk te verklaren door het feit dat Random Forest minder kans op overfitting heeft.

3 Sequenties van activiteiten

3.1 Datacollectie

3.2 Dataverwerking

3.3 Algoritme

3.4 Experimenten en resultaten

4 Conclusie

5 Verder werk