LabelForest: Non-Parametric Semi-Supervised Learning for Activity Recognition

Yuchao Ma, Hassan Ghasemzadeh
School of Electrical Engineering & Computer Science
Washington State University, Pullman WA, 99164, USA
{yuchao.ma, hassan.ghasemzadeh}@wsu.edu

Abstract

Activity recognition is central to many motion analysis applications ranging from health assessment to gaming. However, the need for obtaining sufficiently large amounts of labeled data has limited the development of personalized activity recognition models. Semi-supervised learning has traditionally been a promising approach in many application domains to alleviate reliance on large amounts of labeled data by learning the label information from a small set of seed labels. Nonetheless, existing approaches perform poorly in highly dynamic settings, such as wearable systems, because some algorithms rely on predefined hyper-parameters or distribution models that needs to be tuned for each user or context. To address these challenges, we introduce *LabelForest* [1], a novel non-parametric semi-supervised learning framework for activity recognition. LabelForest has two algorithms at its core: (1) a spanning forest algorithm for sample selection and label inference; and (2) a silhouette-based filtering method to finalize label augmentation for machine learning model training. Our thorough analysis on three human activity datasets demonstrate that LabelForest achieves a labeling accuracy of 90.1% in presence of a skewed label distribution in the seed data. Compared to self-training and other sequential learning algorithms, LabelForest achieves up to 56.9% and 175.3% improvement in the accuracy on balanced and unbalanced seed data, respectively.

1 Introduction

Due to the dynamic nature of human movements and the extreme diversity in data across various users and system settings, developing personalized activity recognition models on mobile sensor systems has remained challenging to date (Yao et al. 2016). In particular, two major limitations of existing machine learning algorithms for activity recognition include (1) the need for adequate labeled training data to achieve high accuracy levels (Longstaff, Reddy, and Estrin 2010); and (2) the lack of flexibility and robustness on diverse data sources.

One promising research direction that attempts to address the first limitation is to combine active learning with semi-supervised learning techniques (Zhu, Lafferty, and Ghahramani 2003). The former takes advantage of the pervasive nature of mobile applications to interact with the user for sensor data labeling. The latter performs label inference from small amounts of initially labeled data samples, henceforth referred to as *"seed"*, to train a machine learning model for automated activity recognition.

Although semi-supervised learning techniques reduce the demand for manual annotations, the limitation imposed due to data diversity remains unsolved. In fact, many semi-supervised learning algorithms attempt to fit in a generative model with a presumed data distribution, or use predefined hyper-parameters as thresholds in data exploration, which relies on *a priori* knowledge about the dataset (Berton and de Andrade Lopes 2015). As a result, these algorithms perform poorly on diverse data sources. In addition, new challenges arise for personalized activity recognition in practice. For example, because it is unlikely to observe all user activities at the same pace, the desired semi-supervised learning algorithms need to be robust to unbalanced seed data with skewed label distribution.

To address the aforementioned challenges, we present the development and validation of LabelForest, a non-parametric and robust semi-supervised learning framework to train personalized activity recognition models. It improves the performance of downstream machine learning model through an *augmentation* approach, which expands the training set by selecting and labeling a subset of available data samples based on an automatic tradeoff between pairwise similarity and accumulative dissimilarity. The experimental results show that, given only one seed data sample per activity, the machine learning model trained with a labeled dataset augmented by LabelForest can achieve an accuracy more than 71.9% on three activity datasets; and it remains an accuracy more than 73.1% when the seed data have highly skewed label distribution.

2 Related Work

Along with the proliferation of sensor technologies and pervasive computing systems, many studies adopted or explored machine learning algorithms on time series sensor data for activity recognition (Kwapisz, Weiss, and Moore 2011; Mannini et al. 2017). Although supervised learning algorithms have demonstrated effectiveness for sensor-based activity recognition, the demand for sufficient amounts of

[1] Software package of LabelForest and sample data are made publicly available at https://github.com/y-max/LabelForest.

labeled data has been identified as a major barrier to user-centric applications (Zakim and Schwab 2015). An encouraging approach to address this challenge is utilizing semi-supervised learning techniques (Chapelle, Scholkopf, and Zien 2009; Ando and Zhang 2005).

Self-training is a straightforward method in semi-supervised learning to perform label augmentation using standard machine learning algorithms. It first trains a weak model with the seed data samples, and then iteratively labels and selects some unlabeled data samples using a confidence measure, to expand the existing training set and re-train the model correspondingly (Cardoso and Moreira 2016). One limitation of self-training is that if a mislabeled data sample is added to the training set in an early stage, the error is propagated through subsequent iterations (Tanha, van Someren, and Afsarmanesh 2017).

Another commonly-used strategy is to apply clustering algorithms on both labeled and unlabeled data samples, so the labels are assigned according to the cluster membership (Zhu 2005). Sequential k-means (Ackerman and Dasgupta 2014; Dias and Cortinhal 2008) is a popular example in this category, by setting the seed data samples as the initial center of the clusters, and gradually adding an unlabeled data sample to its closest cluster. A major limitation of this approach is the sensitivity to outliers, such that the performance varies significantly depending on the input data (Olukanmi and Twala 2017).

Probabilistic models have been used for semi-supervised learning, by fitting data samples with certain assumption about the marginal distribution (Ghazvininejad et al. 2011; Belkin, Niyogi, and Sindhwani 2006). However, the use of unlabeled data samples can adversely impact the performance of the algorithm, if the assumption about the data distribution is not consistent with the truth (Zhu 2005). In particular, we note that the diversity, complexity, and dynamic nature of human movements introduce vast uncertainty in the data gathered by wearable sensors (Rokni and Ghasemzadeh 2017), and such uncertainty makes it infeasible to assume a proper distribution of unknown activity data collected from a specific setting.

Another approach performs label propagation on a graph representation of all the data samples, which is usually constructed using ϵ-Neighborhood (ϵ-NB) or k-Nearest-Neighbor (k-NN) algorithm (Yao et al. 2016). Because both ϵ-NB and k-NN are sensitive to the choice of hyper-parameters (ϵ and k) (Berton and de Andrade Lopes 2015), they require prior knowledge about the data to ensure the validity of the constructed graph. Although a proper graph construction is more important than choosing a label propagation method, less attention has been paid to the former than that of the latter (Zhu 2005).

We introduce a graph-based learning framework that augments labeled dataset through a non-parametric spanning forest construction approach. It does not make prior assumptions on the data distribution, neither require any pre-defined threshold/parameter for neighborhood exploration. As a result, it can improve the performance of machine learning models in a data-driven manner.

3 Problem Statement

This section first defines a sample selection problem in the context of semi-supervised learning for activity recognition, and then presents a graph modeling of this problem and transforms the problem into a spanning forest construction problem. Without loss of generality, we assume that only one seed data sample is given initially for each activity class.

Problem Definition

Due to the dynamic nature of human activities observed with wearable sensing devices, the dataset often contains significant amount of uncertainty and noise, and hence, not all the data samples are informative for activity monitoring. Therefore, LabelForest attempts to select a subset of unlabeled data samples that are substantially similar to the seed data samples, to create a precise training set for model generation.

Problem 3.1 (Sample Selection)**.** *Let $X \in \mathcal{X}$ be a dataset of k-dimensional variables, and a symmetric function $d : X \times X \mapsto \mathbb{R}^+$ defined on $\mathcal{X}$ indicates the pairwise dissimilarity on X. Let $X_Q \subset X$ be a set of M data samples with distinct labels, and $x_i \in X_Q$ is initialized as a cluster C_i, $i \in [1, M]$. The objective of this problem is to select data samples from $\{X \setminus X_Q\}$ to expand the M disjoint clusters, such that for any $x_a \in C_i$, $x_b \in C_j$ and $C_i \neq C_j$, there exists $x_c \in C_i$ and $d(x_a, x_c) \leq d(x_a, x_b)$.*

Graph Modeling

We introduce a graph representation of the data samples to efficiently assess the similarity using the *tree* and *spanning forest* data structures.

Definition 3.1. *Let $G = (V, E)$ be a complete graph derived from a dataset X, where a vertex $v_i \in V$ represents a data sample $x_i \in X$, and an edge $e(v_i, v_j)$ has the weight $w(e_{ij}) \in \mathbb{R}^+$ computed by a distance function $d(x_i, x_j)$. A connected subgraph T on G is said to be a **tree** if it contains no cycles. A set $[T_1, \cdots, T_N]$ on G is called a **spanning forest**, if the vertex set of these trees satisfies $V(T_1) \cup \cdots \cup V(T_N) = V$ and $V(T_i) \cap V(T_j) = \emptyset$ for all $i \neq j$. The construction of $[T_1, \cdots, T_N]$ also suggests a clustering $\mathcal{C} = \{C_1, \cdots, C_N\}$ on X, where $C_i = V(T_i)$ for $i \in [1, N]$.*

Based on the above definition, we can rewrite the Problem 3.1 with the objective of growing a spanning forest on G, given a vertex set $Q \subset V$ with initial labels that are associated with the seed data samples $\{X_Q, Y_Q\}$.

Problem 3.2 (Spanning Forest Construction)**.** *Let $G = (V, E)$ be a complete graph, where $Q \subset V$ is a set of M vertices and each vertex $q_i \in Q$ is considered as the root of the tree T_i, $i \in [1, M]$. This problem aims to construct a spanning forest $F = \{T_1, \ldots, T_N\}$ on G, where $N \leq M$ and F satisfies three conditions: (1) for any $q_i \in V(T_i)$, $q_j \in V(T_j)$ and $q_i \neq q_j$, there is $T_i \neq T_j$; (2) for any $v_a \in V(T_i)$, $v_b \in V(T_j)$ and $T_i \neq T_j$, there exists a $v_c \in V(T_i)$ and $w(e_{ac}) \leq w(e_{ab})$; and (3) if $V(T_k) \cap Q = \emptyset$, then T_k is a single-vertex tree.*

The *smoothness assumption* is widely adopted in graph-based learning algorithms (Zhu 2005), which states that if

two vertices are close to each other on the graph, they are likely to have the same output (i.e., activity label in our case) (Subramanya and Talukdar 2014). Therefore, the spanning forest in the Problem 3.2 presents a natural solution for label inference where the label of the root is shared with all vertices in the same tree.

4 LabelForest Framework Design

LabelForest enhances the performance of machine learning models through two major steps: (1) construct a desirable spanning forest as described in the Problem 3.2 using a greedy spanning forest algorithm introduced in Section 4; and (2) further select highly confident data samples for inclusion in the final training set through a silhouette-based filtering algorithm elaborated in Section 4.

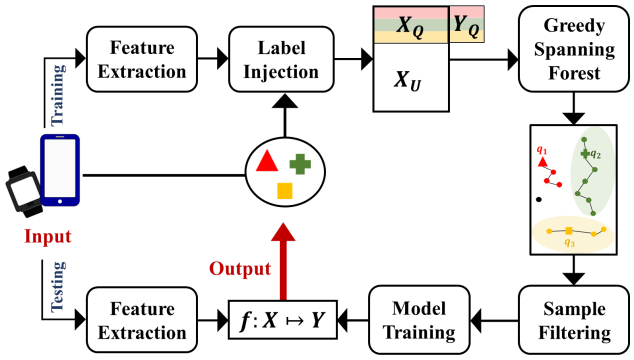


Figure 1: LabelForest framework for personalized activity recognition with limited labels.

Figure 1 shows the workflow using LabelForest for activity recognition on wearable sensor devices. Sensor signals are collected and processed to generate data samples in the form of feature vectors. A few annotations of user activities are provided and attached to the corresponding data samples. LabelForest is then applied on a graph representation of sensor data for label augmentation. Finally, a machine learning model is trained in a supervised manner to carry out activity recognition for future data samples.

Spanning Forest Construction

We design a greedy spanning forest algorithm (GSF) that grows trees rooted by the vertices with initial labels (i.e., seed data samples), and terminates the growth when two trees are about to collide. This algorithm iteratively explores unlabeled data samples represented as vertices, while avoiding the noise introduced by less informative data samples in proximity of the decision boundaries. As a result, this algorithm addresses the Problem 3.2 in a heuristic way, by greedily visiting an unlabeled vertex from a labeled one through an edge that constitutes the smallest weight in each iteration.

In Algorithm 1, the input is a complete graph G with a vertex set $Q \subset V$ representing the seed data samples, and the output is M labeled trees corresponding to M distinct labels for Q. The Euclidean distance is used to measure the weight/distance of each edge $w(e)$. We exploit an automatic

Algorithm 1 Greedy Spanning Forest

Input: $G = (V, E)$ and $Q = \{q_1, \ldots, q_M\} \subset V$
$Accept = Q$; $Reject = \emptyset$; $Margin = \mathbf{0}_M$
$q_i \in Q$ is set as the root of tree T_i
while $\{Accept \cup Reject \neq V\}$ **do**
  $e(u, v)$ = current shortest edge
  if $u, v \notin Accept$ **then**
    $e(u, v)$ = next shortest edge
  end if
  if $u \in V(T_i)$, $v \in V(T_j)$ and $T_i \neq T_j$ **then**
    update $Margin[i]$ and $Margin[j]$ to $w(e)$
  else
    let $u \in V(T_i)$ and $v \notin Accept$
    if $w(e) < Margin[i]$ or $Margin[i] = 0$ **then**
      add v to $Accept$ and $V(T_i)$
    else
      T_k = the tree closest to v
      if $T_k = T_i$ **then**
        add v to $Reject$
      end if
    end if
  end if
end while
Output: $\{T_1, \ldots, T_M\}$ in the spanning forest F.

boundary detection strategy using the concept of *conflicting edge*, to estimate the *margin* of each labeled tree and avoid adding vertices that are less similar to the root due to the accumulative distance.

Definition 4.1 (Conflicting Edge)**.** *Let T_i and T_j be two disjoint trees on the graph G, and their roots have two different labels initially. An edge $e = (u, w)$ is said to be a **conflicting edge** between T_i and T_j if $u \in V(T_i)$ and $w \in V(T_j)$.*

Definition 4.2 (Margin)**.** *If an edge e is a conflicting edge between two trees T_i and T_j, its weight $w(e) \in \mathbb{R}^+$ is said to be the **margin** of T_i and T_j.*

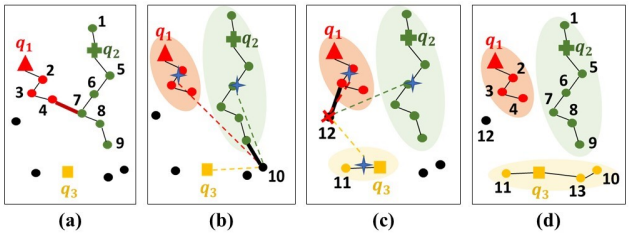


Figure 2: An example of spanning forest construction. Unlabeled vertices are numbered by the visiting order during the spanning process. The shortest edge under examination on each graph is shown as a bold black line. A conflicting edge detected in the process is shown as a bold red line.

Forest Spanning Criteria Algorithm 1 visits the entire vertex set on G by iteratively examining the current shortest edge, which has at least one endpoint belonging to a labeled tree. Let $v \in V(T_i)$ be one endpoint of edge $e = (u, v)$

that is currently under examination, and T_i is a labeled tree rooted by a vertex $q_i \in Q$ with an initial label. There are five possible scenarios for the edge e as follows.

1. If $u \in V(T_i)$, then e forms a cycle and thereby be removed.
2. If $u \in V(T_j)$ and $T_i \neq T_j$, then e is a conflicting edge, and hence, the margin of T_i and T_j is updated according to $w(e)$ (e.g., $e(4,7)$ in Figure 2(a)).
3. If the margin of T_i is 0 or larger than $w(e)$, then add u to $V(T_i)$.
4. If $w(e)$ is not smaller than the margin of T_i, and the distance between u and the center of T_i is smaller than that of other labeled trees, then u is left as a single-vertex tree (e.g., vertex 12 in Figure 2(c)).
5. If $w(e)$ is not smaller than the margin of T_i, but the distance between u and the center of another labeled tree T_k is smaller than that of T_i, then skip making decision for u (e.g., vertex 10 in Figure. 2(b)).

The intuition behind the 4th and 5th criteria is to take into account both the *pairwise similarity* between an unassigned vertex and its closest assigned vertex, and the *accumulative dissimilarity* between that unassigned vertex and the center of labeled trees, which is updated once a new vertex is added into the tree. One thing to note is, a vertex been skipped by the 5th criteria will be re-examined in a later iteration through a different edge. If that edge connects to a labeled tree with the closest center, then this vertex will be added to that labeled tree in the end (e.g., vertex 10 in Figure 2(d)), or left as a single-vertex tree based on the 4th criteria. Because there are finite number of edges on the graph, this algorithm is guaranteed to stop when all the vertices have been assigned to a labeled tree, or rejected from the addition.

Termination of Growth One important property of greedy spanning tree algorithm is to identify those vertices sufficiently similar to the seed data samples in a data-driven manner. This is realized from two aspects: (1) greedily growing a labeled tree through the shortest edge in each iteration, to ensure the intra-similarity within the vertex set of each tree; and (2) terminating the growth of two trees if a conflicting edge between them is detected, to restrict the accumulative distance along the path from the center to any leaf in the tree. The first aspect has been embodied in the initial condition of the algorithm, and we hereby prove the second aspect is guaranteed according to the algorithm design.

Lemma 4.1. *If a conflicting edge e_c is detected between trees T_i and T_j in the greedy spanning forest algorithm, both T_i and T_j stop growing.*

Proof. Let $e_c = (u, v)$ be a conflicting edge, where u and v belong to two disjoint trees T_i and T_j correspondingly, then the margin of T_i and T_j is set to $w(e_c)$. In a later iteration, edge e is the shortest edge among all the edges with at least one endpoint belonging to a labeled tree. If e is adjacent to a vertex in $V(T_i)$ or $V(T_j)$, then there must be $w(e) > w(e_c)$; otherwise, e should be identified as the shortest edge prior to e_c in previous iteration, which contradicts the statement that e is examined after e_c has been detected. Therefore, according to the criterion of $w(e) < w(e_c)$ for adding new vertex to a labeled tree, no more vertex can be added to $V(T_i)$ or $V(T_j)$ once a conflicting edge has been detected. □

Sample Filtering

We propose a silhouette-based filtering algorithm operating on the clusters/forest $\{C_1, \ldots, C_M\}$ constructed by the greedy spanning forest algorithm. The purpose of this step is to perform an automated sample selection for reliable label augmentation.

Silhouette is a popular method widely used for clustering interpretation and validation. It measures the intra-similarity within each cluster versus the inter-dissimilarity between two disjoint clusters (Rousseeuw 1987; Ansari et al. 2011). Given a data sample x_i in a cluster C_i, its silhouette value is estimated as follows.

$$s(x_i) = \frac{inter(x_i) - intra(x_i)}{max\{inter(x_i), intra(x_i)\}} \in [-1, 1] \quad (1)$$

where $inter(x_i)$ denotes the average dissimilarity between x_i and other data samples in C_i, and $intra(x_i)$ denotes the minimum of the average dissimilarity between x_i and data samples in each cluster other than C_i.

Similar to the distance measurement used in the forest spanning, the Euclidean distance is also used to measure the dissimilarity between data samples. According to Eq.1, a higher positive silhouette value indicates a more confident clustering assignment of the data sample, and hence, it is preferred in the sample selection. Algorithm 2 presents the logical flow of our sample filtering process, which aims to remove less confident clustering assignments from the previous step.

Algorithm 2 Silhouette-based Filtering

Input: $C = \{C_1, \ldots, C_M\}$ and $Q = \{q_1, \ldots, q_M\}$
$Center[i] = \mu(C_i)$, $q_i \in C_i$, $i \in [1, M]$
$Base[i] = max\{s(q_i), 0\}$ for C_i
for $x_i \in \{C_i \setminus Q\}$ **do**
  compute $s(x_i)$ using data in $Center \cup Q$
  if $s(x_i) < Base[i]$ **then**
    remove x_i from C_i
    update $Center[i]$
  end if
end for
Output: Clusters $C' = \{C_1, \ldots, C_M\}$

To avoid the use of fixed threshould, Algorithm 2 first computes the silhouette $s(q_i)$ of the seed data sample q_i in the cluster C_i, so that the baseline for sample filtering is the maximum of $\{s(q_i), 0\}$. In the case of only one seed data sample available in one label, the center of the cluster $\mu(C_i)$ is interpolated as an additional data sample to estimate the baseline of silhouette values. Algorithm 2 then examines each data sample assigned by the greedy spanning forest algorithm, and removes the one with a silhouette smaller than the baseline of its assigned cluster. As a result, the filtered

clusters C' have more reliable assignement with low intra-dissimilarity and high inter-dissimilarity. The output of this step is a labeled training set ready for model generation to perform activity recognition.

5 Experiment and Results

We conducted comprehensive analyses to evaluate the performance of LabelForest using three datasets, including (1) *HART* (Anguita et al. 2013; Reyes-Ortiz et al. 2016) contains motion sensor data of 6 daily activities collected from 30 subjects, and the average size of individual data collected from one subject is 347; (2) *SmartSock* contains accelerometer and stretch sensor data of 12 activities collected from 12 subjects, and the average size of individual data is 397; (3) *Phone* (Stisen et al. 2015) contains accelerometer data of 6 activities collected from 9 subjects using a variety of smartphones, and the average size of individual data gathered by one specific smartphone is 1526.

Validation Methodology

Two tasks were designed to evaluate the performance of LabelForest, namely *selective labeling* and *activity recognition*. The first task was to select and label some unlabeled data samples given a few seed data samples. We adopted five metrics to evaluate the performance for this task, and the precision and recall were averaged over all the class labels.

- **labeling rate**: the percentage of data samples been labeled, which reflects the capability of data exploration.
- **precision**: the ratio of true positive to the sum of true positive and false positive.
- **recall**: the ratio of true positive to the sum of true positive and false negative on the data samples been labeled.
- **f1 score**: the weighted sum of precision and recall following the equation below.

$$f1 = \frac{2 \times precision \times recall}{precision + recall}$$

- **labeling accuracy**: the rate of true positive on the data samples been labeled.

The second task was to train a machine learning model using the labeled dataset obtained previously, to estimate the corresponding labels of a separate test set. We empirically chose SVM algorithm to train the classification model for activity recognition, because it achieved better performance across the three datasets comparing to several popular machine learning algorithms. The correct labeling rate on the test set was used to measure the classification accuracy.

Comparison Approaches

Many semi-supervised learning approaches use k-NN or ϵ-NB algorithms to construct a similarity-based graph for label inference (Chapelle and Zien 2005). In our experiment, we compared LabelForest against these two algorithms and other learning approaches utilizing the self-training strategy, as listed below.

- **k-NN**: a number of $k \in \mathbb{Z}$ nearest neighbors of each seed data sample were selected and assigned the same label accordingly.
- **ϵ-NB**: any unlabeled data sample within a given distance $\epsilon \in \mathbb{R}$ to a seed data sample was selected and assigned the same label accordingly.
- **Sequential k-means**: the seed data samples formed an initial clustering according to their labels, and each unlabeled data sample was labeled by comparing its distance to the center of each cluster, which was updated after adding a newly labeled data sample.
- **Naive ML**: A machine learning (ML) model was first trained with the seed data samples, and then updated by re-training with the data samples labeled using the previous model. Three classic machine learning algorithms were tested in the naive approach to provide the baseline performance, including the decision tree (DT), logistic regression (LR) and linear support vector machine (SVM).
- **Upper ML**: the upper bound accuracy of selective labeling is 100% by default. The upper bound accuracy of activity recognition was estimated by training a machine learning model using the true labels of the training set.

Results of Selective Labeling

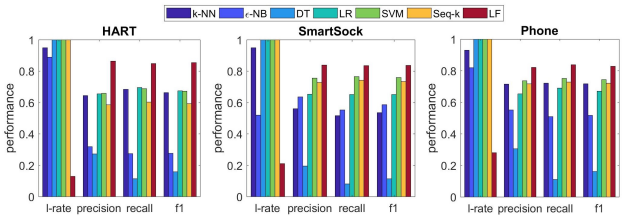


Figure 3: Results of selective labeling given one labeled data sample for each activity class.

Labeling Results Comparison Figure 3 shows the performance of the algorithms for selective labeling given only one seed data sample for each activity class, where "Seq-k" refers to sequential k-means algorithm and "LF" denotes the proposed framework, LabelForest.

Three machine learning algorithms and sequential k-means algorithm assign the labels to all the unlabeled data samples, thereby having the labeling rate of 1. On the contrary, LabelForest gives the priority to the quality of labeling results over the quantity, through an automatic tradeoff without any pre-defined parameter. Although the labeling rate of LabelForest is lower than that of other compared approaches, the precision and recall of LabelForest averaging over all the activity classes is the highest, with an f1 score of 0.86, 0.84 and 0.83 on the three datasets, respectively.

Changes in the Input Data Size We evaluate the performance of each algorithm with changes in the input data size by gradually increasing the number of seed data samples per label from 1 to 6. The results are shown in Figure 4.

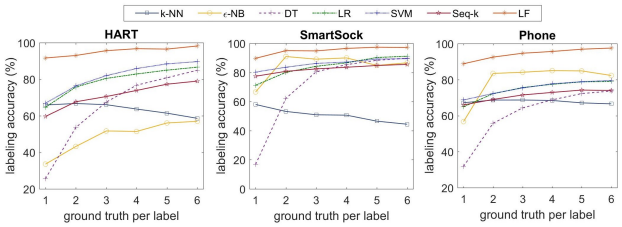


Figure 4: Labeling accuracy with an increasing number of seed data samples per label.

As the number of seed data samples increases, the labeling accuracy of most algorithms improve accordingly, which is attributed to a better labeling model learned from a more informative dataset. However, k-NN and ϵ-NB algorithms show the fluctuations in their performance, mainly because of the randomly selected parameter values. Overall speaking, LabelForest not only achieves the highest labeling accuracy (more than 88.8% in all the test cases), but also shows a steady improvement along with the increasing number of seed data samples, with an accuracy increase of 7.2%, 8.3% and 9.8% on the three datasets respectively.

Results of Activity Recognition

The second task of our experiment is activity recognition using the augmented training set obtained in previous task. The baseline performance is computed by training the model with seed data samples alone, and then testing it on a separate test set. Figure 5 shows the results of activity recognition with an evenly growing number of seed data samples.

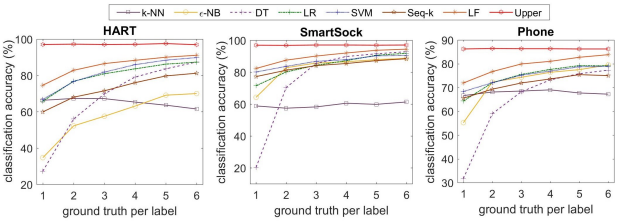


Figure 5: Accuracy of activity recognition with increasing number of seed data samples.

The upper bound is computed as mentioned in Section 5, and the overall accuracy appears to be significantly lower on the *Phone* dataset than that of the other two datasets, where the average accuracy is 86.3% for the *Phone* dataset, 97.1% for the *HART* dataset and 96.9% for the *SmartSock* dataset. It indicates the diversity of human activity data caused by the differences in individual movement patterns and sensing systems.

Overall speaking, the performance of LabelForest is the closest to the upper bound performance comparing to other tested algorithms. The classification accuracy of LabelForest increases from 74.6% to 91.1% on the *HART* dataset, from 82.3% to 94.4% on the *SmartSock* dataset, and from 71.9% to 83.8% on the *Phone* dataset, along with the increase in the number of seed data samples.

Impact of Unbalanced Seed Dataset

In the previous experiment, the number of seed data samples increased evenly for all the activity classes. However in practice, it is more likely to observe uneven numbers of seed data samples for different activities. To validate the robustness of LabelForest given a seed dataset with skewed label distribution, we obtain an unbalanced seed dataset by adding more seed data samples in only one randomly picked label, while maintaining the number of seed data samples to one for the other labels in the validation.

Labeling with Skewed Input We first compare the performance of each algorithm for selective labeling given balanced and unbalanced seed dataset, as shown in Figure 6, where "balanced L2" means each activity class has 2 data samples in the seed dataset, and "unbalanced L8" means, in the seed dataset, one random activity class has 8 data samples and the others have only 1 data sample for each. We choose these two cases for comparison because the total size of the seed dataset is similar.

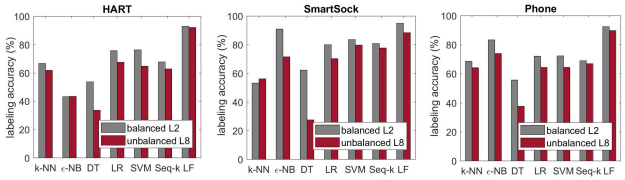


Figure 6: Performance of selective labeling with balanced and unbalanced seed datasets.

In general, almost all the algorithms show a decline in the labeling accuracy given unbalanced seed dataset, but the performance of LabelForest has a less significant change comparing to other algorithms, with an accuracy decrease of 3.6% averaging over the three datasets. This result demonstrates the robustness of LabelForest in presence of a skewed label distribution in the seed dataset. In addition, LabelForest also achieves the best labeling performance in the case of unbalanced seed dataset, with an accuracy of 92.2%, 88.5% and 89.8% on the three datasets respectively.

Activity Recognition with Skewed Input We further evaluate the performance of the machine learning model trained with the augmented training set, when the seed dataset has an increasing degree of the skewness, which is simulated by gradually increasing the number of seed data samples for only one activity class but remaining only one data sample for the other activity classes. The results are shown in Figure 7.

The results in this figure have less changes comparing to the results in Figure 5, mainly because only one activity class has an increasing number of data samples in the seed dataset for the former. It shows that the additional information regarding to only one class cannot contribute significantly to the labeling performance, and hence, the augmented training set cannot be representative enough to improve the performance of down-stream machine learning model. Overall speaking, LabelForest still has the most

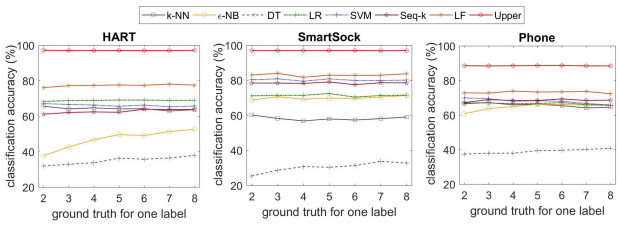
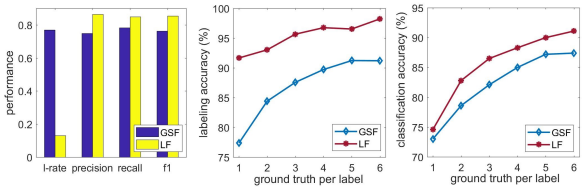


Figure 7: Accuracy of activity recognition with the increasing skewness in the seed dataset.

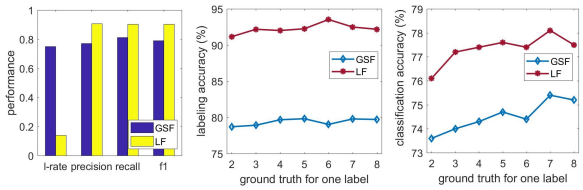
improvement in the performance of the machine learning model through training set augmentation, and the classification accuracy is 77.3% on the first dataset, 83.0% on the second dataset and 73.1% on the third dataset.

Impact of Sample Filtering

As presented in Section 4, LabelForest consists of two major steps, the first one performs a greedy spanning forest (GSF) algorithm to explore the data samples similar to the seed data samples, and the second step uses a silhouette-based filtering algorithm to finalize the newly labeled data samples based on the first step. To validate the effectiveness of these two steps, we compare the performance of LabelForest on the *HART* dataset with and without applying the second step of sample filtering.



(a) Comparison with balanced seed dataset



(b) Comparison with highly unbalanced seed dataset.

Figure 8: Performance comparison of LabelForest with and without applying sample filtering.

Figure 8 shows the comparison results given balanced (Figure 8a) and unbalanced (Figure 8b) seed datasets, where "GSF" refers to the case of using greedy spanning forest algorithm alone, and "LF" refers to the case of performing the LabelForest framework completely. The motivation of sample filtering is to further improve the quality of the newly labeled data samples by trading the labeling rate for the precision, which is reflected in the two bar plots in Figure 8. Given balanced seed dataset with one seed data sample for each activity class, the labeling rate of LF is 16.6% as that of GSF, but the precision of LF is 15.4% higher than that of GSF. In the case of unbalanced seed dataset with 8 seed data samples for only one activity class, the labeling rate of LF is 18.8% as that of GSF, while the precision of LF is 17.5% higher than that of GSF. As a result, although the labeled dataset obtained after sample filtering is smaller in size than that of GSF, the higher precision of the former results in a more accurate machine learning model for activity recognition, as shown in the rightmost plots in Figure 8.

6 Conclusions and Future Work

The dynamic nature of human movements requires the underlying computational algorithm to be robust to handle the diversity of individual data distributions. In practice, collecting sufficiently large amounts of labeled training data is time consuming and expensive. Therefore, it is desirable to autonomously construct a large training dataset using an initially small labeled dataset.

In this study, we designed and developed an autonomous label augmentation framework for semi-supervised learning, called *LabelForest*, to obtain effective activity recognition models in wearable sensing systems. Our approach learns from a small set of seed data with initial activity labels, and develops novel graph-based sample selection and label inference algorithm coupled with a silhouette-based filtering strategy. The experimental results demonstrated the robustness of LabelForest with a labeling accuracy of 90.1% on average.

Many semi-supervised learning algorithms use a label propagation method on the constructed graph to assign labels to unlabeled data samples. Because our graph construction aims to obtain a spanning forest by giving a higher priority to the labeling precision over the labeling rate, the resulting graph is naturally a disconnected graph. Therefore, existing label inference/propagation algorithms cannot be applied directly on the graph obtained by LabelForest. If there is large overlap between data samples of two different classes, a conflicting edge will be easily detected in an early stage during forest spanning, which terminates the growth of the two trees/clusters consequently. As a result, a small amount of data samples will be labeled in this case. An interesting future work that we plan to pursue is to refine our graph construction algorithm, such that the existing label propagation algorithms can be directly applied for label inference.

A limitation of the LabelForest framework presented in this article is that it assumes the labeled and unlabeled data samples are available in batch. Therefore, the graph construction, forest formation, labeling, and sample selection are all performed in a static or off-line fashion. We note that it is desirable if the semi-supervised learning can be performed in real-time as new sensor data become available. As part of our future work, we plan to design incremental learning strategies in the context of semi-supervised learning for real-time motion analysis in wearable sensing systems.

7 Acknowledgments

This work was supported in part by the United States National Science Foundation, under grant CNS-1750679. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding organizations.

References

Ackerman, M., and Dasgupta, S. 2014. Incremental clustering: The case for extra clusters. *CoRR* abs/1406.6398.

Ando, R. K., and Zhang, T. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.* 6:1817–1853.

Anguita, D.; Ghio, A.; Oneto, L.; Parra, X.; and Reyes-Ortiz, J. L. 2013. A public domain dataset for human activity recognition using smartphones. In *ESANN*.

Ansari, Z. A.; Azeem, M. F.; Ahmed, W.; and Babu, A. V. 2011. Quantitative evaluation of performance and validity indices for clustering the web navigational sessions. *CoRR* abs/1507.03340.

Belkin, M.; Niyogi, P.; and Sindhwani, V. 2006. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *J. Mach. Learn. Res.* 7:2399–2434.

Berton, L., and de Andrade Lopes, A. 2015. Graph construction for semi-supervised learning. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, IJCAI'15.

Cardoso, H. L., and Moreira, J. M. 2016. Human activity recognition by means of online semi-supervised learning. In *2016 17th IEEE International Conference on Mobile Data Management (MDM)*, volume 2, 75–77.

Chapelle, O., and Zien, A. 2005. Semi-supervised classification by low density separation. In *AISTATS 2005*, 57–64. Max-Planck-Gesellschaft.

Chapelle, O.; Scholkopf, B.; and Zien, A. 2009. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks* 20(3):542–542.

Dias, J. G., and Cortinhal, M. J. 2008. The skm algorithm: A k-means algorithm for clustering sequential data. In Geffner, H.; Prada, R.; Machado Alexandre, I.; and David, N., eds., *Advances in Artificial Intelligence – IBERAMIA 2008*, 173–182. Berlin, Heidelberg: Springer Berlin Heidelberg.

Ghazvininejad, M.; Rabiee, H. R.; Pourdamghani, N.; and Khanipour, P. 2011. Hmm based semi-supervised learning for activity recognition. In *Proceedings of the 2011 International Workshop on Situation Activity & Goal Awareness*, SAGAware '11, 95–100. New York, NY, USA: ACM.

Kwapisz, J. R.; Weiss, G. M.; and Moore, S. A. 2011. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter* 12(2):74–82.

Longstaff, B.; Reddy, S.; and Estrin, D. 2010. Improving activity classification for health applications on mobile devices using active and semi-supervised learning. In *2010 4th International Conference on Pervasive Computing Technologies for Healthcare*, 1–7.

Mannini, A.; Rosenberger, M.; Haskell, W. L.; Sabatini, A. M.; and Intille, S. S. 2017. Activity recognition in youth using single accelerometer placed at wrist or ankle. *Medicine and science in sports and exercise* 49(4):801.

Olukanmi, P. O., and Twala, B. 2017. Sensitivity analysis of an outlier-aware k-means clustering algorithm. In *2017 Pattern Recognition Association of South Africa and Robotics and Mechatronics (PRASA-RobMech)*, 68–73.

Reyes-Ortiz, J.-L.; Oneto, L.; Samà, A.; Parra, X.; and Anguita, D. 2016. Transition-aware human activity recognition using smartphones. *Neurocomput.* 171(C):754–767.

Rokni, S. A., and Ghasemzadeh, H. 2017. Synchronous dynamic view learning: A framework for autonomous training of activity recognition models using wearable sensors. In *2017 16th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 79–90.

Rousseeuw, P. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* 20(1):53–65.

Stisen, A.; Blunck, H.; Bhattacharya, S.; Prentow, T. S.; Kjærgaard, M. B.; Dey, A.; Sonne, T.; and Jensen, M. M. 2015. Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, SenSys '15, 127–140. New York, NY, USA: ACM.

Subramanya, A., and Talukdar, P. P. 2014. *Graph-Based Semi-Supervised Learning*. Morgan & Claypool Publishers.

Tanha, J.; van Someren, M.; and Afsarmanesh, H. 2017. Semi-supervised self-training for decision tree classifiers. *International Journal of Machine Learning and Cybernetics* 8(1):355–370.

Yao, L.; Nie, F.; Sheng, Q. Z.; Gu, T.; Li, X.; and Wang, S. 2016. Learning from less for better: Semi-supervised activity recognition via shared structure discovery. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '16, 13–24. New York, NY, USA: ACM.

Zakim, D., and Schwab, M. 2015. Data collection as a barrier to personalized medicine. *Trends in pharmacological sciences* 36.

Zhu, X.; Lafferty, J.; and Ghahramani, Z. 2003. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *ICML 2003 workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, 58–65.

Zhu, X. 2005. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison.