TimeNet: Pre-trained deep recurrent neural network for time series classification

Pankaj Malhotra MALHOTRA.PANKAJ@TCS.COM
Vishnu TV VISHNU.TV@TCS.COM
Lovekesh Vig LOVEKESH.VIG@TCS.COM
Puneet Agarwal PUNEET.A@TCS.COM
Gautam Shroff GAUTAM.SHROFF@TCS.COM
TCS Research, New Delhi, India

Abstract

In the spirit of the tremendous success of deep Convolutional Neural Networks as generic feature extractors from images, we propose *Timenet*: a multilayered recurrent neural network (RNN) trained in an unsupervised manner to extract features from time series[1]. Fixed-dimensional vector representations or embeddings of variable-length sentences have been shown to be useful for a variety of document classification tasks. Timenet is the encoder network of an auto-encoder based on sequence-to-sequence models that transforms varying length time series to fixed-dimensional vector representations. Once Timenet is trained on diverse sets of time series, it can then be used as a *generic off-the-shelf feature extractor for time series*. We train Timenet on time series from 24 datasets belonging to various domains from the UCR Time Series Classification Archive, and then evaluate embeddings from Timenet for classification on 30 other datasets not used for training the Timenet. We observe that a classifier learnt over the embeddings obtained from a pre-trained Timenet yields significantly better performance compared to (i) a classifier learnt over the embeddings obtained from the encoder network of a domain-specific auto-encoder, as well as (ii) a nearest neighbor classifier based on the well-known and effective Dynamic Time Warping (DTW) distance measure. We also observe that a classifier trained on embeddings from Timenet give competitive results in comparison to a DTW-based classifier even when using significantly smaller set of labeled training data, providing further evidence that Timenet embeddings are robust. Finally, t-SNE visualizations of Timenet embeddings show that time series from different classes form well-separated clusters.

1. Introduction

Event detection from sensor data is of great importance to engineers who are concerned with heavy industrial equipments such as cranes, water pumps, and pulverizers, for making critical design, engineering and operational decisions. The current industry practice is to use unsupervised heuristic driven domain-based models for such tasks. Rampant use of diagnostic trouble codes in automobile industry, Mahalanobis Taguchi method for anomaly detection in power-plants (Taguchi et al., 2000), and many other purpose specific heuristics such as a deceleration exceeding half of gravitational acceleration being termed as "hard stop" in vehicles (Hassan et al., 2015) are examples of such industry practices. Such approaches remain in mainstream use primarily because of unavailability of labeled time series data. However, with the advent of "Industrial Internet", unlabeled time series data from sensors is available in abundance. In this paper, we intend to explore the possibility of

1. This is an extended version of the work published as a conference paper in the Proceedings of 25th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning.

learning a model from the unlabeled time series data, which could be used to transform time series to *representations* that are useful for further analysis, and call this model *Timenet*.

Learning good representations from data is an important task in machine learning (Bengio et al., 2013). Recently, fixed-dimensional vector representations for words (Mikolov et al., 2013) and for variable length sequences of words in the form of sentences, paragraphs, and documents have been successfully used for natural language processing tasks such as machine translation and sentiment analysis (Le and Mikolov, 2014; Kiros et al., 2015). Noticeably, deep Convolutional Neural Networks trained on millions of images from 1000 object classes from ImageNet (Russakovsky et al., 2015) have been used as off-the-shelf feature extractors to yield powerful generic image descriptors for a diverse range of tasks such as image classification, scene recognition and image retrieval (Sharif Razavian et al., 2014; Simonyan and Zisserman, 2014). These features or representations have even been shown to outperform models heavily tuned for the specific tasks.

Deep recurrent neural networks (RNNs) have been shown to perform hierarchical processing of time series with different layers tackling different time scales (Hermans and Schrauwen, 2013; Malhotra et al., 2015). This serves as a motivation to explore a multilayered RNN as a Timenet model that processes and transforms a univariate time series to a fixed-dimensional vector representation, and once trained on diverse enough time series, serves as a generic off-the-shelf feature extractor for time series. To learn such a Timenet in an unsupervised manner, we leverage sequence-to-sequence (seq2seq) models (Cho et al., 2014; Sutskever et al., 2014): a seq2seq model consists of an encoder RNN and a decoder RNN. The encoder RNN takes a sequence as input and encodes it to a fixed-dimensional vector representation given by the final hidden state of the encoder, and the decoder RNN decodes this vector representation to generate another sequence as output. We train a seq2seq model on time series of varying length from diverse domains, and once trained, freeze the encoder RNN to be used as Timenet (refer Section 3 for details). Seq2seq models have been proposed for time series modeling tasks such as audio segment representation (Chung et al., 2016), anomaly detection from sensor data (Malhotra et al., 2016a), and determining health of machines from sensor data (Malhotra et al., 2016b).

To evaluate Timenet, we compare the embeddings obtained from Timenet with the embeddings given by the encoder RNN of a domain-specific SAE model on several time series classification (TSC) datasets from UCR Time Series Classification Archive (Chen et al., 2015). For many datasets not used for training the Timenet, the embeddings from Timenet yield better classification performance compared to embeddings from the encoder of an SAE trained specifically on the respective datasets, and also outperforms a Dynamic Time Warping (DTW) based nearest neighbor classifier (DTW-C) which is a popular benchmark for TSC, and has been shown to be competitive across domains (Ding et al., 2008). We also perform qualitative analysis of the embeddings from Timenet using t-SNE (Van der Maaten and Hinton, 2008).

The contributions of this paper can be summarised as follows:

1. We show that it is possible to train *Timenet: a deep recurrent neural network that serves as an off-the-shelf generic feature extractor for time series*. For several time series classification datasets which are not used for training Timenet, the classifiers learnt over embeddings given by Timenet perform better compared to i) classifiers over embeddings from data-specific RNNs learnt in same manner as the Timenet, and ii) DTW-C (Section 4.3).
2. Even when using *significantly lesser amount of labeled training data*, Timenet embeddings based classifier gives competitive classification performance compared to DTW-C, suggesting that the embeddings are robust (Section 4.3).
3. *We show that Timenet captures important characteristics of time series.* Timenet based embeddings of time series belonging to different classes form well-separated clusters when visualized using t-SNE (Section 4.2). Similarly, Timenet based embeddings of time series from datasets belonging to different domains are also well-separated.

The rest of the paper is organized as follows: We briefly introduce multilayered (deep) RNNs in Section 2. Next, we explain our approach for learning Timenet using seq2seq models in Section 3, followed by detailed experimental evaluation of Timenet and comparison with data-specific encoders in Section 4. We discuss the related work in Section 5, and conclude in Section 6 with observations and possible future directions.

2. Preliminary: Multilayered RNN with Dropout

We briefly introduce deep RNNs with Gated Recurrent Units (Cho et al., 2014) in the hidden layers with dropout for regularization (Pham et al., 2014; Zaremba et al., 2014). Dropout is applied to non-recurrent connections ensuring that the state of any hidden unit is not affected. This is important in RNNs to allow information flow across time-steps. For lth hidden layer of a multilayered RNN with L layers, the hidden state $\mathbf{h}_t^l$ is obtained from the previous hidden state $\mathbf{h}_{t-1}^l$ and the hidden state $\mathbf{h}_t^{l-1}$ of the layer $l-1$. The hidden state transition from $t-1$ to t for layer l is given by function f:

$$f : \mathbf{h}_t^{l-1}, \mathbf{h}_{t-1}^l \rightarrow \mathbf{h}_t^l \tag{1}$$

The function f is implemented through the following transformations iteratively for $t = 1$ to T:

$$reset\,gate : \mathbf{r}_t^l = \sigma(\mathbf{W}_r^l.[\mathbf{D}(\mathbf{h}_t^{l-1}), \mathbf{h}_{t-1}^l]) \tag{2}$$

$$update\,gate : \mathbf{u}_t^l = \sigma(\mathbf{W}_u^l.[\mathbf{D}(\mathbf{h}_t^{l-1}), \mathbf{h}_{t-1}^l]) \tag{3}$$

$$proposed\,state : \tilde{\mathbf{h}}_t^l = \tanh(\mathbf{W}_p^l.[\mathbf{D}(\mathbf{h}_t^{l-1}), \mathbf{r}_t \odot \mathbf{h}_{t-1}^l]) \tag{4}$$

$$hidden\,state : \mathbf{h}_t^l = (1 - \mathbf{u}_t^l) \odot \mathbf{h}_{t-1}^l + \mathbf{u}_t^l \odot \tilde{\mathbf{h}}_t^l \tag{5}$$

where $\odot$ is Hadamard product, $[\mathbf{a}, \mathbf{b}]$ is concatenation of vectors $\mathbf{a}$ and $\mathbf{b}$, $\mathbf{D}(.)$ is dropout operator that randomly sets the dimensions of its argument to zero with probability equal to dropout rate, $\mathbf{h}_t^0$ is the input z_t at time-step t. $\mathbf{W}_r$, $\mathbf{W}_u$, and $\mathbf{W}_p$ are weight matrices of appropriate dimensions s.t. $\mathbf{r}_t^l, \mathbf{u}_t^l, \tilde{\mathbf{h}}_t^l$, and $\mathbf{h}_t^l$ are vectors in $\mathbb{R}^{c^l}$, where c^l is the number of units in layer l. The sigmoid (σ) and *tanh* activation functions are applied element-wise.

3. Learning Timenet using Sequence Auto-encoder

We consider a sequence auto-encoder (SAE) based on seq2seq models (refer Figure 1). The SAE consists of two multilayered RNNs with Gated Recurrent Units in the hidden layers (as introduced in Section 2): an encoder RNN and a decoder RNN. The encoder and decoder RNNs are trained jointly to minimize the reconstruction error on time series from diverse domains in an unsupervised manner. Once such an SAE is learnt, the encoder RNN from the encoder-decoder pair is used as a pre-trained Timenet model to obtain embeddings for test time series.

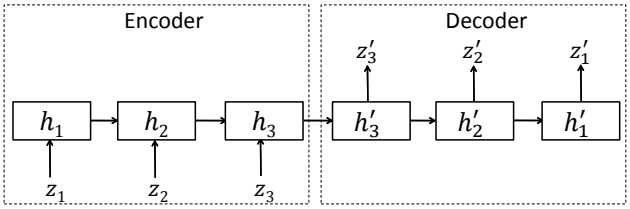


Figure 1: SAE for sample time series $\{z_1, z_2, z_3\}$. The final encoder state (h_3) is used as the embedding of the time series.

More specifically, given a time series $Z^{(i)} = \{z_1^{(i)}, z_2^{(i)}, ..., z_{T^{(i)}}^{(i)}\}$ of length $T^{(i)}$, $\mathbf{h}_t^{(i)}$ is the hidden state of the encoder at time t, where $\mathbf{h}_t^{(i)} \in \mathbb{R}^c$. The number of hidden units in the encoder is

$c = \sum_{l=1}^{L} c^l$. The encoder captures information relevant to reconstruct the time series as it encodes the time series, and when it reaches the last point in the time series, the hidden state $\mathbf{h}_{T^{(i)}}^{(i)}$ is the vector representation or embedding for $Z^{(i)}$. The decoder has the same network structure as the encoder, with the final hidden state $\mathbf{h}_{T^{(i)}}^{(i)}$ of the encoder being used as the initial hidden state of decoder. The decoder additionally has a linear layer as the output layer. The decoder reconstructs the time series in reverse order, similar to (Sutskever et al., 2014), i.e., the target time series is $\{z_{T^{(i)}}^{(i)}, z_{T^{(i)}-1}^{(i)}, ..., z_1^{(i)}\}$. The SAE is trained to reconstruct the input time series so as to minimize the objective $E = \sum_{i=1}^{N} \sum_{t=1}^{T^{(i)}} (z_t^{(i)} - z_t'^{(i)})^2$, where $z_t'^{(i)}$ is the reconstructed value corresponding to $z_t^{(i)}$, N is the number of time series instances.

Unlike the conventional way of feeding an input sequence ((Cho et al., 2014)) to the decoder during training as well as inference, the only inputs the decoder gets in our SAE model are the final hidden state of the encoder, and the steps T for which the decoder has to be iterated in order to reconstruct the input. We observe that the embeddings or the final encoder states thus obtained carry all the relevant information necessary to represent a time series (refer Section 4).

Time complexity of obtaining embeddings from Timenet: Consider a Timenet model with L hidden layers, with each layer having c recurrent units (without loss of generality). For any hidden layer $l > 1$ and time-step t, the operations mentioned in Equations 2 - 4 involve multiplication of a weight matrix of dimensions $c \times 2c$ with a hidden state vector of size $2c \times 1$. For $l = 1$, the weight matrix having dimension $c \times (c + 1)$ is multiplied with vector of size $c + 1$. Therefore, the number of computations across layers is $O(Lc^2)$. Equation 5 involves element-wise product of c-dimensional vectors, and hence involves $O(Lc)$ operations. Therefore, for a time series of length T, the total number of computations are $O(Lc^2T)$, which is linear in time series length T for a given Timenet architecture.

4. Experimental Evaluation

We begin with the details of how Timenet and data-specific SAEs were trained in Section 4.1; and then present a qualitative analysis of the ability of Timenet to learn robust embeddings using t-SNE in Section 4.2. We compare Timenet embeddings with data-specific SAE embeddings on several TSC datasets in Section 4.3. The time series datasets used are taken from UCR TSC Archive (Chen et al., 2015), which is a source of large number of univariate time series datasets belonging to diverse domains. We also evaluate TimeNet on an industrial telematics dataset consisting of hourly readings from six sensors installed on engines in vehicles, where the task is to classify normal and abnormal behavior of engines (referred as Industrial Multivariate in Fig. 2a and Table 1).

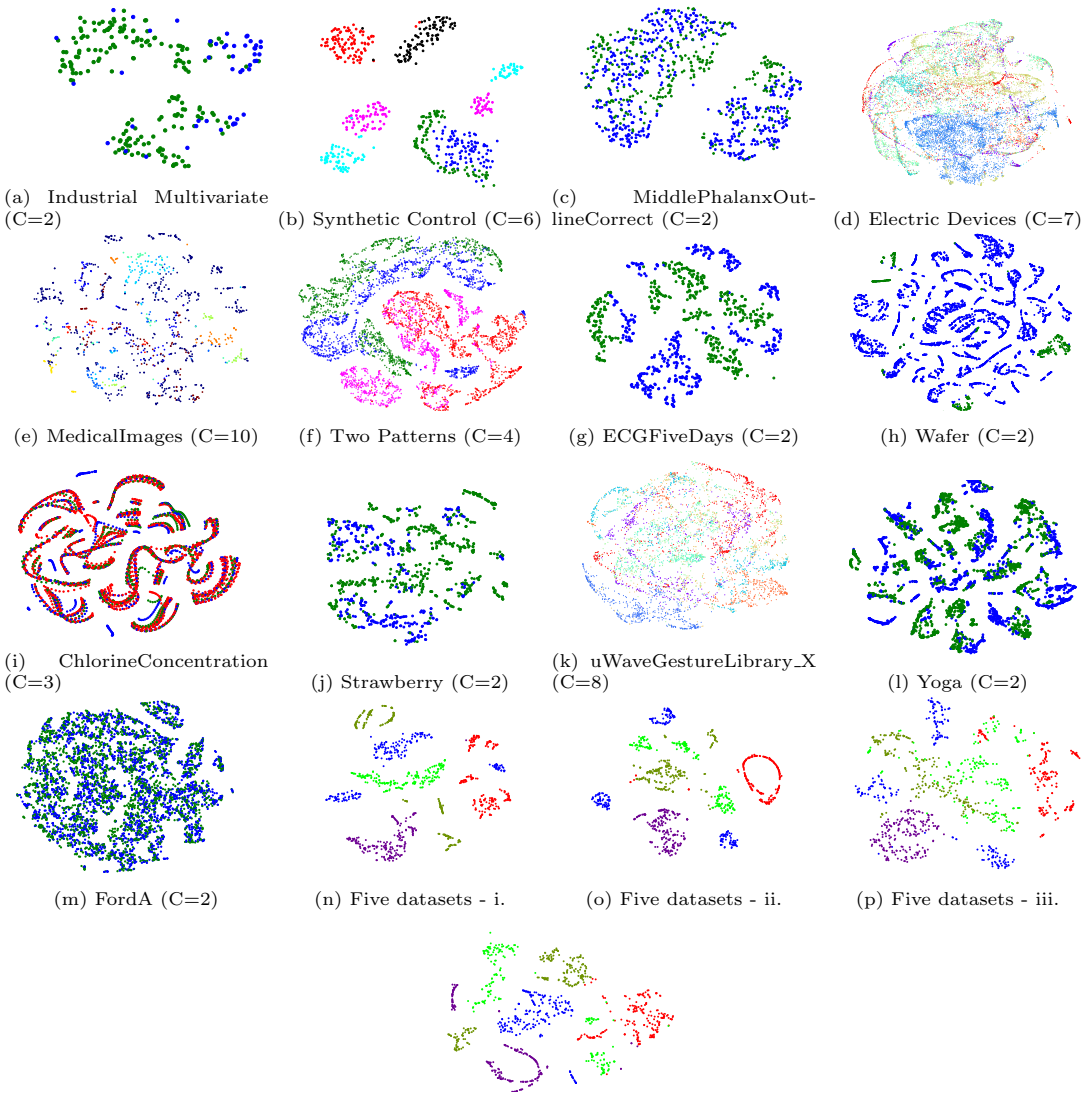
4.1 Timenet and data-specific SAE training details

We chose 18 datasets for training, 6 datasets for validation, and another 30 as test datasets for Timenet evaluation from the UCR TSC Archive. The datasets are chosen such that time series length $T \leq 512$. Each dataset comes with a pre-defined train-test split and a class label for each time series. The training dataset is diverse as it contains time series belonging to 151 different classes from the 18 datasets with T varying from 24 to 512 (refer Table 2 for details). When using a dataset as part of training or validation set, all the train and test time series from the dataset are included in the set. Each time series is normalized to have zero mean and unit variance.

We use Tensorflow (Abadi et al., 2015) for implementing our models, and use Adam optimizer (Kingma and Ba, 2014) for training. We use learning rate 0.006, batch size 32, and dropout rate 0.4. All hidden layers have same number of hidden units (c^l). The best architecture is selected to have minimum average reconstruction error on the time series in the validation set. The best Timenet model obtained has $c^l = 60$ and $L = 3$ such that the embedding dimension $c = 180$. This model takes about 17 hours to train for about 7k iterations on a NVIDIA Tesla K40C 12GB GPU.

Data-specific SAE models are trained under same parameter settings while tuning for c^l and L, in an unsupervised manner, using only the training set of the respective dataset.

4.2 Visualization of embeddings using t-SNE



(a) Industrial Multivariate (C=2)
(b) Synthetic Control (C=6)
(c) MiddlePhalanxOutlineCorrect (C=2)
(d) Electric Devices (C=7)
(e) MedicalImages (C=10)
(f) Two Patterns (C=4)
(g) ECGFiveDays (C=2)
(h) Wafer (C=2)
(i) ChlorineConcentration (C=3)
(j) Strawberry (C=2)
(k) uWaveGestureLibrary_X (C=8)
(l) Yoga (C=2)
(m) FordA (C=2)
(n) Five datasets - i.
(o) Five datasets - ii.
(p) Five datasets - iii.
(q) Five datasets - iv.

Figure 2: Timenet embeddings visualized using t-SNE. Image best viewed magnified.

Figures 2a - 2m show t-SNE visualizations of embeddings obtained from Timenet for test datasets (refer Table 1) with number of classes $C \leq 10$ (for visual clarity). For each test dataset, the embeddings for all the time series from both train and test splits are shown. Due to space constraints, we show t-SNE visualization for one dataset each (selected randomly) from the ten Phalanx datasets and two ECG datasets (as listed in Table 1). To compare the embeddings obtained from Timenet across test datasets, we consider 200 randomly selected time series each from five randomly selected test datasets, and visualize them using t-SNE. Figures 2n - 2q show t-SNE plots for four such random

selections. We observe that embeddings for time series belonging to different classes within a dataset form well-separated clusters in most cases (Figures 2a - 2m). Further, the embeddings for time series from one dataset are well-separated from embeddings for time series from other datasets (Figures 2n - 2q). This suggests that *Timenet is able to capture important characteristics of time series in the embeddings produced.*

4.3 Embeddings from Timenet and data-specific SAE

For each test dataset, we learn two non-linear Support Vector Machine (SVM) (Burges, 1998) classifiers with radial basis function (RBF) kernel: i) TN-C: using Timenet embeddings as features, ii) SAE-C: using embeddings obtained from the encoder of a data-specific SAE model as features. It is to be noted that none of the 30 test datasets on which we evaluate Timenet are used for training the Timenet. However, for training a data-specific SAE we use the training set of the respective dataset (without using the class label information). The parameters C (cost) and γ of the RBF kernel of SVM are selected using 5-fold cross-validation over the logarithmic grid of 10^{-3} to 10^{3}. The classification error rates for TN-C, SAE-C, and DTW-C are reported in Table 1. The error rates for DTW-C are taken from (Lines and Bagnall, 2015).

Dataset	**T**	**DTW-C**	**SAE-C**	**TN-C**	**TN-$C_{2/3}$**	**TN-C_{L1}**	**TN-C_{L2}**	**TN-C_{L3}**
Industrial Multivariate	30	-	0.221	**0.173**	0.176	0.135	0.154	0.154
Synthetic Control	60	0.017	0.017	**0.013**	0.016	0.010	0.013	0.027
PhalangesOutlinesCorrect	80	0.239	0.228	**0.207**	0.225	0.213	0.221	0.217
DistalPhalanxOutlineAgeGroup*	80	0.228	**0.160**	0.223	0.211	0.178	0.200	0.165
DistalPhalanxOutlineCorrect*	80	0.232	**0.187**	0.188	0.201	0.188	0.178	0.185
DistalPhalanxTW*	80	0.272	0.243	**0.208**	0.220	0.203	0.213	0.223
MiddlePhalanxOutlineAgeGroup*	80	0.253	0.348	**0.210**	0.229	0.215	0.280	0.205
MiddlePhalanxOutlineCorrect*	80	0.318	0.307	**0.270**	0.344	0.475	0.472	0.295
MiddlePhalanxTW*	80	0.419	0.381	**0.363**	0.392	0.361	0.371	0.366
ProximalPhalanxOutlineAgeGroup*	80	0.215	**0.137**	0.146	0.154	0.141	0.151	0.156
ProximalPhalanxOutlineCorrect*	80	0.210	0.179	**0.175**	0.199	0.175	0.175	0.175
ProximalPhalanxTW*	80	0.263	**0.188**	0.195	0.194	0.200	0.200	0.188
ElectricDevices	96	0.376	0.335	**0.267**	0.288	0.265	0.280	0.309
MedicalImages	99	0.253	**0.247**	0.250	0.271	0.238	0.246	0.232
Swedish Leaf	128	0.157	**0.099**	0.102	0.139	0.123	0.126	0.115
Two Patterns	128	0.002	0.001	**0.000**	0.002	0.000	0.002	0.007
ECG5000	140	0.075	**0.066**	0.069	0.069	0.063	0.069	0.066
ECGFiveDays*	136	0.203	**0.063**	0.074	0.150	0.129	0.127	0.096
Wafer	152	**0.005**	0.006	**0.005**	0.007	0.008	0.006	0.009
ChlorineConcentration	166	0.35	0.277	**0.269**	0.344	0.227	0.250	0.314
Adiac	176	0.391	0.435	**0.322**	0.372	0.366	0.304	0.294
Strawberry	235	**0.062**	0.070	**0.062**	0.075	0.090	0.072	0.077
Cricket_X	300	**0.236**	0.341	0.300	0.321	0.346	0.326	0.364
Cricket_Y*	300	**0.197**	0.397	0.338	0.363	0.379	0.351	0.400
Cricket_Z*	300	**0.180**	0.305	0.308	0.336	0.328	0.338	0.359
uWaveGestureLibrary_X	315	0.227	**0.211**	0.214	0.228	0.219	0.216	0.220
uWaveGestureLibrary_Y*	315	0.301	**0.291**	0.311	0.326	0.304	0.307	0.335
uWaveGestureLibrary_Z*	315	0.322	**0.280**	0.281	0.295	0.298	0.289	0.286
Yoga	426	**0.155**	0.174	0.160	0.200	0.176	0.152	0.173
FordA	500	0.341	0.284	**0.219**	0.229	0.234	0.242	0.261
FordB	500	0.414	0.405	**0.263**	0.285	0.263	0.299	0.298
Win or Tie compared to DTW-C		-	22/30	**25/30**	20/30	22/30	22/30	21/30

Table 1: Classification error rates. Here, TN-C_{Li} is the classifier learnt using embeddings from ith layer of Timenet. * means SAE from the first dataset in the group was used for training SAE-C.

Amongst the test datasets, there are four cases where multiple datasets belong to the same domain. In such cases, we train one SAE model per domain using the dataset with the largest number of train instances (refer Table 1). However, the classifier SAE-C is trained on the respective training set. We observe that TN-C and SAE-C exceed or match the performance of DTW-C on 83% (25/30) datasets and 73% (22/30) datasets, respectively. Also, TN-C exceeds the performance of SAE-C on 60% (18/30) datasets. These results further confirm that *a pre-trained Timenet gives*

embeddings which provide relevant features for TSC. TN-C is better than the recently proposed state-of-art PROP (proportional ensemble of classifiers based on elastic distance measures) on 4 out of 15 test datasets for which the results have been reported in (Lines and Bagnall, 2015).

TN-C with reduced labeled training data: We further test the robustness of embeddings by reducing the amount of labeled data used, and learn a classifier TN-$C_{2/3}$ using randomly selected two-thirds of the training data while maintaining class balance. We report average of the error rate of three such random selections in Table 1. TN-$C_{2/3}$ performs better than DTW-C on 66% (20/30) datasets suggesting that *robust embeddings can be obtained using a pre-trained Timenet, and then a classifier can be learnt over the embeddings using lesser amount of labeled information.*

Performance of each layer of Timenet: To evaluate the relevance of each layer of Timenet, we learn SVM classifier TN-C_{Li} using embeddings from only the ith hidden layer. We observe that for datasets with small T, a single layer of Timenet gives reasonable classification performance, and at times performs better than TN-C. For datasets with large T, the classifier TN-C is better than any TN-C_{Li} (refer Table 1). This suggests that *for shorter time series, one of the three layers extracts relevant features from time series, whereas for longer time series all layers carry relevant information.*

5. Related Work

Many approaches for TSC analyzed in (Lines and Bagnall, 2015) use variations of time warping and edit distance in a nearest neighbor classifier. Other approaches extract statistical and frequency-based features from time series, and learn a classifier over these features. To the best of our knowledge, our work is the first to show that it is possible to leverage unlabeled varying length time series from diverse domains to train a multilayered recurrent network as a feature extractor.

Recurrent neural networks (RNNs) for TSC have been proposed in (Hüsken and Stagge, 2003). Several approaches for time series modeling using deep learning based on models such as RNNs, conditional Restricted Boltzmann Machines (RBM), temporal RBMs, Time-Delay Neural Networks, and Convolutional Auto-encoder have been surveyed in (Längkvist et al., 2014). (Längkvist et al., 2012) proposed unsupervised feature learning using Deep Belief Networks (DBNs) for sleep stage classification from time series. The static models such as Convolutional auto-encoder and DBNs partition the time series into windowed time series of fixed length to analyze temporal data. Our approach proposes the seq2seq model which uses a pair of RNNs for learning representations without need of windowing, and may be more naturally suited for temporal data related tasks. Also, to the best of our knowledge, ours is the first approach which proposes a generic unsupervised pre-trained RNN as time series encoder for diverse univariate time series.

Very recently, (Oord et al., 2016) showed that it is possible to have one generic deep neural network for raw audio waveforms that can capture different characteristics of many different speakers with equal fidelty. (Chung et al., 2016) proposed Audio Word2Vec model where they show usefulness of learning fixed-dimensional representations from varying length audio signals using seq2seq models. Our work extends the above approaches specific to speech domain, and shows that it is possible to learn a generic encoder for time series from diverse domains. Semi-supervised sequence learning proposed in (Dai and Le, 2015) shows that LSTMs pre-trained using sequence auto-encoders on unlabeled data are usually better at classification than LSTMs initialized randomly. This work uses the encoder network to initialize the LSTM classifier whereas we use the embeddings obtained from encoder for classification. Data augmentation techniques (e.g. Yadav et al. (2015), Le Guennec et al. (2016)) have been proposed to handle data scarcity for training deep models for sequences or time series. ODEs as a generative model for time series have been shown to improve performance of RNNs for anomaly detection Yadav et al. (2015). Convolutional Neural Networks for TSC (Le Guennec et al., 2016) has been recently proposed where data augmentation through window slicing and warping is proposed to handle scarcity of data. Further, pre-training of each layer using auto-encoder is proposed using similar-length time series from other domains (data mixing) to improve

performance. Our work is different from this approach in the sense that we use pre-trained encoder network learnt on diverse time series of varying length rather than leveraging slicing, warping, and data mixing from similar-length time series to handle scarcity of data for training deep networks.

6. Discussion

Deep neural networks are data-intensive models. In practical applications, getting labeled data is costly although unlabeled data may be readily available. Recently, sequence-to-sequence models trained in an unsupervised manner as auto-encoders have been successfully demonstrated for spoken term detection from audio segments, and for anomaly detection and prognostics from sensor data. We exploit such a sequence auto-encoder trained on diverse unlabeled time series data to obtain a deep recurrent neural network (RNN) that transforms time series to fixed-dimensional representations or embeddings. This pre-trained multilayered RNN is found to yield effective embeddings for time series classification. The t-SNE visualizations further confirm the ability of Timenet to yield meaningful embeddings. The encoders of domain-specific sequence auto-encoders also provide useful representations for time series classification although they may be prone to overfitting given few training instances. Studying the usefulness of embeddings obtained from Timenet for other tasks such as anomaly detection and clustering is a plausible direction for future work.

References

Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, et al. Tensorflow: Large-scale machine learning on heterogeneous systems, 2015. *tensorflow.org*, 1, 2015.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

Yanping Chen, Eamonn Keogh, Bing Hu, Nurjahan Begum, et al. The ucr time series classification archive, July 2015. `www.cs.ucr.edu/~eamonn/time_series_data/`.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, et al. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Yu-An Chung, Chao-Chung Wu, Chia-Hao Shen, and Hung-Yi Lee. Unsupervised learning of audio segment representations using sequence-to-sequence recurrent neural networks. *arXiv preprint arXiv:1603.00982*, 2016.

Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems*, pages 3079–3087, 2015.

Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552, 2008.

Ehtesham Hassan, Gautam Shroff, and Puneet Agarwal. Multi-sensor event detection using shape histograms. In *Proceedings of the Second ACM IKDD Conference on Data Sciences*, pages 20–29. ACM, 2015.

Michiel Hermans and Benjamin Schrauwen. Training and analysing deep recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 190–198, 2013.

Michael Hüsken and Peter Stagge. Recurrent neural networks for time series classification. *Neurocomputing*, 50:223–235, 2003.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL `http://arxiv.org/abs/1412.6980`.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, et al. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302, 2015.

Martin Längkvist, Lars Karlsson, and Amy Loutfi. Sleep stage classification using unsupervised feature learning. *Advances in Artificial Neural Systems*, 2012:5, 2012.

Martin Längkvist, Lars Karlsson, and Amy Loutfi. A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognition Letters*, 42:11–24, 2014.

Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. In *ICML*, volume 14, pages 1188–1196, 2014.

Arthur Le Guennec, Simon Malinowski, and Romain Tavenard. Data augmentation for time series classification using convolutional neural networks. In *ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data*, 2016.

Jason Lines and Anthony Bagnall. Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, 29(3):565–592, 2015.

Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks for anomaly detection in time series. In *23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2015.

Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Lstm-based encoder-decoder for multi-sensor anomaly detection. In *Anomaly Detection Workshop at 33rd International Conference on Machine Learning (ICML 2016)*. CoRR, abs/1607.00148, 2016, https://arxiv.org/abs/1607.00148, 2016a.

Pankaj Malhotra, Vishnu TV, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Multi-sensor prognostics using an unsupervised health index based on lstm encoder-decoder. In *1st ACM SIGKDD Workshop on Machine Learning for Prognostics and Health Management, San Francisco, CA, USA, 2016*. CoRR, abs/1607.00148, 2016. URL http://arxiv.org/abs/1607.00148, 2016b.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, et al. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

Vu Pham, Théodore Bluche, Christopher Kermorvant, and Jérôme Louradour. Dropout improves recurrent neural networks for handwriting recognition. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 285–290. IEEE, 2014.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 806–813, 2014.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*. 2014.

Genichi Taguchi, Yuin Wu, and Subir Chodhury. *Mahalanobis-Taguchi System*. McGraw-Hill Professional, 2000. ISBN 0071362630.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85, 2008.

Mohit Yadav, Pankaj Malhotra, Lovekesh Vig, K Sriram, and Gautam Shroff. Ode-augmented training improves anomaly detection in sensor data from machines. In *NIPS Time Series Workshop*. CoRR, abs/1605.01534. URL http://arxiv.org/abs/1605.01534, 2015.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.

Appendix A. Datasets used for training Timenet

Dataset	T	C	N	Dataset	T	C	N
ItalyPowerDemand	24	2	1096	**SonyAIBORobotSurfaceII**	65	2	980
SonyAIBORobotSurface	70	2	621	**TwoLeadECG**	82	2	1162
FacesUCR	131	14	2250	**Plane**	144	7	210
Gun_Point	150	2	200	**ArrowHead**	251	3	211
WordSynonyms	270	25	905	**ToeSegmentation1**	277	2	268
Lightning7	319	7	143	**ToeSegmentation2**	343	2	166
DiatomSizeReduction	345	4	322	**OSULeaf**	427	6	442
Ham	431	2	214	**Fish**	463	7	350
ShapeletSim	500	2	200	**ShapesAll**	512	60	1200

(a) Training datasets

Dataset	T	C	N	Dataset	T	C	N
MoteStrain	84	2	1272	**CBF**	128	3	930
Trace	275	4	200	**Symbols**	398	6	1020
Herring	512	2	128	**Earthquakes**	512	2	461

(b) Validation datasets

Table 2: Training and validation datasets used for Timenet. Here, T: time series length, C: number of classes, N: number of time series.