D2D-LSTM: LSTM-Based Path Prediction of Content Diffusion Tree in Device-to-Device Social Networks*

Heng Zhang,[1] Xiaofei Wang,[1]† Jiawen Chen,[1] Chenyang Wang,[1] Jianxin Li[2]
[1]Tianjin Key Laboratory of Advanced Networking, College of Intelligence and Computing, Tianjin University, China
[2]Deakin University
841636@gmail.com, {xiaofeiwang, chenyangwang}@tju.edu.cn,
cjw602001243@163.com, jianxin.li@deakin.edu.au

Abstract

With the proliferation of mobile device users, the Device-to-Device (D2D) communication has ascended to the spotlight in social network for users to share and exchange enormous data. Different from classic online social network (OSN) like Twitter and Facebook, each single data file to be shared in the D2D social network is often very large in data size, e.g., video, image or document. Sometimes, a small number of interesting data files may dominate the network traffic, and lead to heavy network congestion. To reduce the traffic congestion and design effective caching strategy, it is highly desirable to investigate how the data files are propagated in offline D2D social network and derive the diffusion model that fits to the new form of social network. However, existing works mainly concern about link prediction, which cannot predict the overall diffusion path when network topology is unknown. In this article, we propose D2D-LSTM based on Long Short-Term Memory (LSTM), which aims to predict complete content propagation paths in D2D social network. Taking the current user's time, geography and category preference into account, historical features of the previous path can be captured as well. It utilizes prototype users for prediction so as to achieve a better generalization ability. To the best of our knowledge, it is the first attempt to use real world large-scale dataset of mobile social network (MSN) to predict propagation path trees in a top-down order. Experimental results corroborate that the proposed algorithm can achieve superior prediction performance than state-of-the-art approaches. Furthermore, D2D-LSTM can achieve 95% average precision for terminal class and 17% accuracy for tree path hit.

Introduction

With the increasing popularity of smart devices, people prefer to download various files from network, and duplicated resource downloading leads to the explosion of cellular network traffic and huge waste of resources. Cha (Cha et al. 2007; Cha et al. 2009) confirmed that 10% of popular videos account for 80% of views in YouTube, and caching a small number of contents can effectively solve the problem of repeated resources downloading. Device-to-Device(D2D) technology can effectively cache these resources. As for caching, Hao(Hao, Zhang, and Tao 2018), Gregori(Gregori et al. 2016) and Ramy(Amer et al. 2018) proposed some strategies on content caching, but their mathematical hypotheses show much deviation in real application. Furthermore, Anderson (Anderson 1998) confirmed that customer satisfaction is related to word of mouth, while Chevalier(Chevalier and Mayzlin 2006) proved that a book's good reviews can lead to a increase in relative sales at that Amazon due to word-of-mouth effect. These two facts prove the great influence of people meeting and sharing like D2D communication on content diffusion. The principle of D2D communication is shown in Fig1.

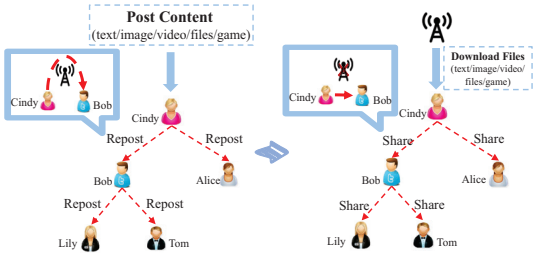

Figure 1: Working principle of D2D sharing

In this paper, we concentrate on how representative APPs[1] are propagated in offline D2D social network. Specifically, every APP belongs to one category, while one's preference for APPs reflects his interest, personality, social habit and so on. We can learn that a file's spread potential is not only limited to the content type itself but the file's owner's preference for time, geographical location and so on. Therefore, we designed our D2D-LSTM to capture the current user's properties with the history of users who previously shared the APP to predict this file's future propagation path. To further adapt

*This work is partially supported by the National Key R&D Program of China (2018YFC0809803, 2019YFB2101901), China NSFC (Youth) through grant 61702364, China NSFC GD Joint fund U1701263, ARC Linkage Project LP180100750.

†Xiaofei Wang is corresponding author.

[1]APP refers to application software designed to run on smartphones and other mobile devices.

D2D-LSTM to new users, individual users are clustered into a series of prototype users in terms of their feature vectors. Each prototype user can represent a group of similar users. In general, the prototype user is used for training and making prediction, while we use the real D2D content propagation trace to train and evaluate the optimization model. The main contributions of this paper can be summarized as follows:

- Our work is the first to model D2D content propagation with multivariate features, e.g., content, time, geographical location, transmission preference and so on. Also, we're the first to learn the tree path with a Long Short-Term Memory (LSTM) neural network.
- Our work is the first to propose a Tree-to-Sequence algorithm to give D2D-LSTM the ability for learning and predicting a complete tree path in D2D social network. Experimental results on the real-world dataset demonstrate that D2D-LSTM can significantly improve the effectiveness of at most 39.2% mAP increase and at most 88% tree hit rate increase.
- D2D-LSTM is more generalizable and robust in tree generation for a given root node. It only relies on prototype users which are more available, reliable and robust than precise users widely-used by state-of-the-art approaches in the propagation predicting process.

Method

Problem Statement

We formulate tree propagation path prediction as a decision-making process. There is an agent perceiving the environment through sensors of the neutral network and performing a series of actions in the environment through actuators, then a series of actions in the environment are performed through actuators, so as to optimize a goal. Let $\mathcal{A} = \{a_1, a_2, ..., a_{|A|}\}$ represent a series of APPs, and let $\mathcal{U} = \{u_1, u_2, ..., u_{|U|}\}$ represent a set of prototype users. In tree propagation path prediction, the goal is to generate a sequence of diffusion path $P(a_i) = \{(\omega_1, 1), (\omega_2, 2), ..., (\omega_t, t), ...\}$ for a given an APP file a_i,in which tuple (ω_t, t) indicate that a given APP is delivered to prototype user ω_t at time step t. Without loss of generality, we set the time of an APP firstly being propagated to 0. The purpose of this paper is to use the propagation paths before time t to predict the prototype users involved during time t. We use $AP_{terminal}$(Average Precision for the terminal users), mAP_{real}(mean Average Precision for all real prototype users), mAP_{all}(mean Average Precision for all prototype users), Kappa coefficient(used for the consistency test) and macro-F1(used to measure the classification effect in the case of multi-labels) to measure the effectiveness of D2D-LSTM.

Trace Data Analysis

We set up our experiments with large-scale real-world datasets from a worldwide D2D file sharing APP. This APP provides users the ability to share contents across platforms (e.g Windows, Android, iOS), through WiFi-Direct, Bluetooth etc, without 3G/4G/LTE cellular network infrastructure. We build a Spark&Hadoop big data processing platform to process the large-scale trace. Eventually, the data columns after preprocessing are (File Type, MD5, Sender ID, Receiver ID, Timestamp, Country, GPS, File size). Detailed statistic of the trace is shown in Table 1.

Table 1: Trace Statistic

Statistic Item	Value
Users	30485335
Records Count	4434440043
Files	16785175
Time Range	From 2016.08.01 To 2016.10.30
Trace Total Size	90GB

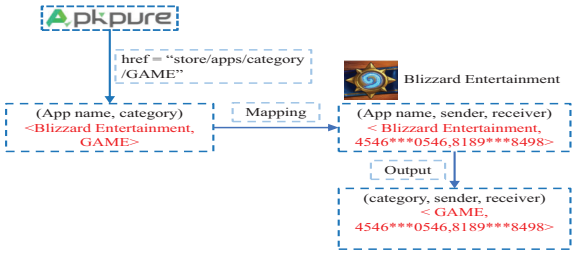

Figure 2: APP classification process

In order to get user's preference for APPs, APPs' name and type from Apkpure are collected to classify the APPs transmitted in the trace. Each APP corresponds to a specific category. Taking Blizzard Entertainment as an example, Fig 2 shows the detailed classification process. APPs are grouped into 48 types including Music & Audio, Education, Social and so on. And the distrubution of these 48 categories is shown in Fig 3.

As can be seen from Fig 3, different categories of APP in offline D2D social network has a different number of transmissions. Top 10 hot APP categories are shown in Table 2. Video Players & Editors is the most popular type among those APP categories, accounting for 17% or so. The Tools category followed, with 11% of users choosing to share such category of APPs. While Art & Design accounted for the lowest percentage, only less than 1%.

Table 2: Top 10 hot APPs in the trace

Category Name	Percent	Category Name	Percent
Video Players & Editors	16.66%	Action	5.61%
Tools	10.72%	Business	5.41%
Communication	9.28%	Racing	4.82%
Productivity	7.41%	Board	4.23%

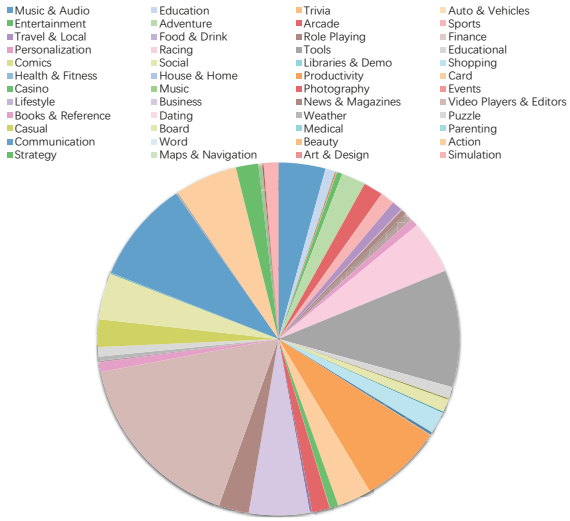

Figure 3: 48 types of APP from our trace

User Prototype Modeling

A user's history trace is used to generate his feature vector. Every record in trace is mapped to a one-hot feature vector including 48-dimensional content category vector, 24-dimensional time vector, T-dimensional geographical position vector (The value of T is discussed later) and 2-dimensional sending times and receiving times vector. Generate user feature vector by superimposing and regularizing one-hot vectors of the same user.

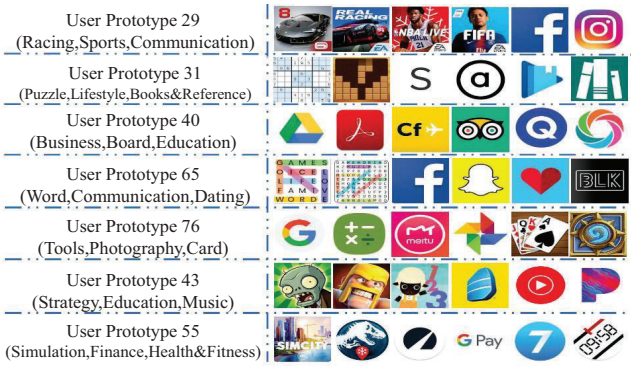

Figure 4: Example of Prototype User

All possible diffusion paths depend on the users in the MSN as well as the social network structure itself (e.g., the connections between users). Therefore, we need to generalize a set of similar users into prototype users to represent the general shared features, and map each user to prototype user based on his preference(such as the APP categories they like to spread, propagation times and the area in which they are often shared). This has been verified in previous work (Liu et al. 2016) that size of the spread trees of some APPs are limited, and the depth is about 4, so using prototype user is necessary. In this way, we can ignore the nuances of real users during training and testing, and regard predicting prototype user as a training target. **K-means** is used to describe the social characteristics of users.

Users are clustered into 80 prototypes (Cluster size's influence on prediction performances will be discussed in later chapters), Fig 4 describes the examples of the prototype users. The ID of prototype user and his favorite three categories are shown in the left panel, and the right panel shows six typical APPs belonging to the three categories. For example, prototype user 43 prefers to share strategy games, education and music APPs, and they often send games like plants vs. zombies, temple escapes, etc.

Tree Path Building and Transformation

We separate the D2D communication activities trace by APP name and generate the propagation path forest for every APP. In the construction of tree path propagation, we take propagations' time sequence into consideration. Algorithm 1 shows the construction procedure of the tree propagation path. To evaluate the complexity of the algorithm, we divide the process into two steps. In the first step, every log entry in the log set is traversed, and this step's complexity is $O(n)$. We find the sender in vertex set and utilize Red Black Tree to optimize the indexing speed in the second step, so this step's complexity is $O(lg(n))$. The whole algorithm's complexity is $O(nlgn)$, where n is the length of the log set.

Algorithm 1: tree path construction

Input : $LogSet$ = Set(Row(TimeStamp, Sender, Receiver, OtherProps))
Output: Tree structure in memory
$RootSet = \emptyset$, $VertexSet = \emptyset$
Sort($LogSet$, Sorted in ascending order by TimeStamp)
foreach $log_i \in LogSet$ **do**
  V_s = Find $Sender(log_i)$ in $Vertex$
  if V_s = *Not found* **then**
    V_s = Create node $Sender(log_i)$
    Add V_s to $VertexSet$
    Add V_s to $RootSet$
  end
  V_r = Find $Receiver(log_i)$ in $Vertex$
  if V_r = *Not found* **then**
    V_r = Create node $Reciever(log_i)$
    Add the V_r to set $VertexSet$
  end
  Add V_r to $V_s.Childs$
end

LSTM is taken to learn and predict the propagation path, so we convert tree paths into sequence paths so the trees can adapt to LSTM's input. Similar to Zhang (Zhang, Lu, and Lapata 2015), we use traversal algorithm to make such conversion, and the sequence paths serve as the input of D2D-LSTM, thus we can solve the problem of horizontal expanding during training and evaluation. To address the uncertainty of the propagation trees' breadth and depth, a virtual

node is added to represent the ending of the current depth, which can be used to learn the vertical propagation of the tree.

The traversal order in the transformation procedure determines the learning order of LSTM. In this paper, Breadth-First (BFS) traversal is adopted to traverse the propagation tree. The BFS order reflects the hierarchical relationship of nodes in the tree and takes the parent-child relationship in the propagation process into account. In the transformation process, the corresponding sibling node and parent node of a node are recorded. Noting that the feature vector of the virtual node is randomly generated.

Algorithm 2: propagation tree to sequence

Input : tree struture $ForestArray$
Output: Sequence node list $ResultSet$
initialization List $Q = \emptyset$, $Sibling = \emptyset$, $ResultSet = \emptyset$
foreach $T \in ForestArray$ **do**
  Add root node T to Q
  repeat
    node = first node in Q
    foreach *tnode∈ children list of node* **do**
      tnode is added to the end of the Q
      Add tuple (node, Sibling, tnode)to $ResultSet$
      Sibling = tnode
    end
    Add tuple (node, V-NODE)
  until Q *is* $\emptyset$;
end

The specific transformation process is described in algorithm 2, where V-NODE means added virtual node. There are three nested loops in the algorithm. For the first loop, it traverses all root nodes in the forest, so its complexity is $O(m)$ where m is the number of root nodes in the forest. In the second loop, all nodes in a tree will be traversed so its complexity is $O(p)$, where p is less than the max nodes count n for trees. In the third loop, every child of the parent node in Q will be traversed, so the complex is $O(q)$ where q is less than the maximum number of children for every node. Because there might be chain path in forest, so $m < n$, $p < n$, $q < n$, so the algorithm's complexity is $O(n^3)$. Fig 5 shows an example of a tree path transformed into a sequence path.

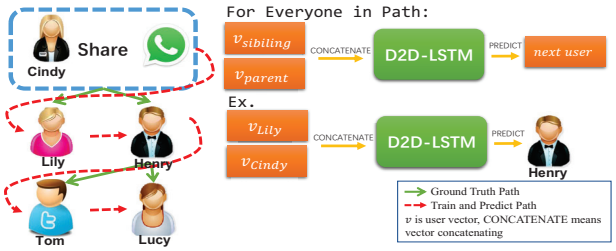

Figure 5: Example of Tree Path Transformation

LSTM Network Architecture

Apart from current properties, the historical diffusion path plays a key role in deciding whether the APP will be further propagated, and if so, to whom next. This is a recursive problem that is well suited for a recursive neural network (RNN) to analyze the diffusion path. However, RNN suffers from gradient explosion of weights and the disappearance of gradients. To address these problems, we used LSTM to train our model(Gers, Schmidhuber, and Cummins 1999; Sundermeyer, Schlüter, and Ney 2012) instead of the RNN(Graves, Mohamed, and Hinton 2013).

In our work, all transmission histories of an APP are memorized due to the memory nature of LSTM. The architecture of D2D-LSTM is illustrated as Fig 6. The memorized histories are used to predict the future propagation path of the file. Previous memorized cell state and hidden state in time step $t-1$ are passed to LSTM Cell in time step t. At time step t, the parent node's feature vector and the left sibling node's feature vector are concatenated to one vector, and this vector is put in the LSTM Cell, so do the hidden state and cell state at time $t-1$. The output h and c are propagated to the next LSTM Cell, and an FC layer with a sigmoid layer follow the output y to map the hidden vector to specific prototype user. The output from each sub moment makes up a Prototype sequence and the sequence can be recovered to the tree path with virtual child nodes. The D2D-LSTM equation is expressed as follows:

$$\begin{aligned} i_t &= \sigma(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)}) \\ f_t &= \sigma(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}) \\ o_t &= \sigma(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}) \\ u_t &= \sigma(W^{(u)}x_t + U^{(u)}h_{t-1} + b^{(u)}) \\ c_t &= i_t \odot u_t + f_t \odot c_{t-1} \\ h_t &= o_t \odot \tanh(c_t) \end{aligned} \quad (1)$$

where f_t is the input social feature of the user at the current step, W, U and b are the weight matrices and bias vectors to be optimized, and $\odot$ denotes element-wise multiplication. f_t represents the forget gate vector, the input gate vector is signed by i_t , o_t denotes an output gate vector, the memory cell can be written as c_t, and the hidden state is denoted as h_t.

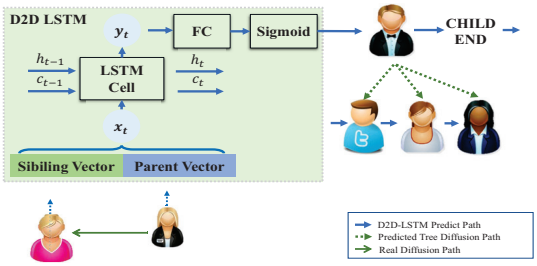

Figure 6: The architecture of D2D-LSTM model

Loss Function

The tree prediction problem can be treated as BFS-order nodes generation and the generation of each node can be regarded as a classification problem. So we model the process of generating a new node as a multi-binary classification and we can calculate confidence for every class and select the prototype user which has the maximum confidence. Then we optimize our model via multi-class binary cross-entropy loss:

$$L_{log}(W) = -\frac{1}{N}\sum_{i=0}^{N-1}\sum_{k=0}^{K-1} y_{i,k}\, logp_{i,k} \qquad (2)$$

where W is a set of weight matrices and N represents the number of training samples and K is the number of label. The probability of the currently predicted node belonging to prototype user k is $p_{i,k}$,and $y_{i,k}$ is the label of prototype user k.

Experiments

We perform substantial experiments to evaluate D2D-LSTM and compare it with other baselines. The baselines involved in this experiment include FC with all four features, D2D-LSTM with random weights, D2D-LSTM without the feature of content category, time feature, geographical position, and time preference respectively, so as to verify the predictive accuracy performance of D2D-LSTM. In addition, this paper demonstrates the effect of features mentioned above by comparison.

Implementation Details

In this paper, PyTorch is used to construct and implement a specific neural network. In deep learning, gradient descent is adopted to update network weights. We use the initial learning rate of 0.008. When loss increases compared with the previous step, the learning rate decreases by half. Adam was taken in the optimization model, with the first-order moment estimated attenuation rate of 0.9 and the second-order moment estimated attenuation rate of 0.999. In LSTM training, mini-batch gradient descent was carried out, and the mini-batch size was 64. As the length of ground-truth tree varies, the padding method is employed for alignment, and the completed elements are ignored in the calculation process. For all training models, the maximum number of iterations epoch is set to 300.

Baselines

We compare our D2D-LSTM model with the following baselines:

RNN with all four features: It has the same architecture as that of D2D-LSTM, but uses general RNN to learn and predict. The purpose of this model is to verify the importance of memory during the path prediction process.

D2D-LSTM with random weights: It has the same architecture as our D2D-LSTM model but with random weights. This is to test the random performance ability of D2D-LSTM, so as to compare with the performance of D2D-LSTM after training.

FC Model with all four features: Same to D2D-LSTM, this baseline has all four features. FC Model contains only one hidden layer and one Softmax layer to convert the output hidden layer vector into the probability of being propagated this time. The purpose of this model is to verify the importance of memory in the transmission process.

D2D-LSTM Model lack of one feature: It has the same architecture as that of D2D-LSTM, but only lack of one feature respectively. To ensure dimensional consistency, the missing feature is filled with 0. These baselines are used to test the importance of missing dimension in communication.

Table 3 shows the configuration of baselines, where ✓means use this feature and × means not.

Table 3: Baseline Configuration

ID	Model	Content	Time	Loc	Trans
#1	FC	✓	✓	✓	✓
#2	Random Weight	✓	✓	✓	✓
#3	D2D-LSTM	✓	✓	✓	✓
#4	D2D-LSTM	×	✓	✓	✓
#5	D2D-LSTM	✓	×	✓	✓
#6	D2D-LSTM	✓	✓	×	✓
#7	D2D-LSTM	✓	✓	✓	×
#8	RNN	✓	✓	✓	✓

D2D-LSTM is a top-down tree path prediction and generation, but the Tree-LSTM (Zhang, Lu, and Lapata 2015) aims to perform a bottom-up tree data classification. So we cannot compare it to Tree-LSTM.

Node-wise Prediction Accuracy

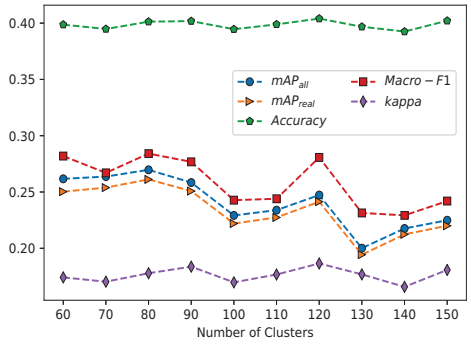

Figure 7: The variation trend of measurement in different T

Due to the limitation that only one node can be output at a time in the LSTM, D2D-LSTM can just be predicted node by node. Therefore, prototype user predicted by D2D-LSTM and real prototype user will be used to make a node by node comparison. Then, we calculated the average precision of the predictions for each prototype, and used macro

evaluation F1-Score and Kappa coefficient to measure the confidence coefficient and consistency of the overall prediction category. Limited to node-by-node prediction, we did not generate tree paths at this stage. To further simplify the model and improve the generalization of D2D-LSTM, this paper proposes to divide users into T prototypes according to their social characteristics. Fig 7 shows the variation trend of measurement index in different T. mAP_{all} is the mean average precision (AP) for overall prototype user classes, including terminal class, while mAP_{real} is the mean average precision for all real prototype user classes, Which does not contain terminal class. From this figure, it can be concluded that D2D-LSTM performs best when T = 80. So we choose T = 80 to train our model. When T = 80, the accuracy of prediction, macro-F1, Kappa coefficient, mAP_{real} and mAP_{all} are 39.17% 26.19%, 17.14%, 23.00% and 23.88%, respectively.

Fig 8 shows the performance evaluation of D2D-LSTM, and compares it to other baselines. The accuracy of RW is 0, and its loss is very large, so it is not shown in the figure. mAP is the mean AP computed over all 81 classes. $AP_{terminal}$ is the average precision for terminal class. And AP_{real} represents the mean AP for all prototype users.

First, we can see that all baseline models but RW and RNN outperform D2D-LSTM with random weights baseline, which produces a very low mAP_{real} showing the difficulty of the task. The fact that the LSTM model is higher than the mAP of the RNN model indicates that adding the forgotten gate and memory mechanism will improve the accuracy of the prediction, thus verifying the effectiveness of using LSTM as a basis in this paper. And $AP_{terminal}$ of all models is higher than mAP of them. Reason for this phenomenon is that the terminal class has a 1 : 2 ratio of non-leaf vs. leaf (total of 65172, 22351 non-leaf and 42821 leaf). The large difference in target ratios make the prediction task for the prototype classes much more difficult than the terminal class.

Second, we conclude the social characteristics of users playing an essential role in D2D transmission path prediction. Comparing D2D-LSTM Model lack of different feature, it can be found that geographical location plays an important role in D2D communication, because D2D transmission is mainly constrained by geographical location according to its characters.

Third, our D2D-LSTM performs much better than the FC model. The accuracy and mean average precision of D2D-LSTM with all features increase 8.0% and 39.2% at most respectively. This phenomenon indicates the significance of memorizing APP transfer histories in D2D-LSTM cell for predicting the next forward user.

Finally, as can be seen from the result, D2D-LSTM converges faster, LOSS decreases faster and our loss (0.83) is the minimum among other baselines. Meanwhile, this model has a maximum Macro-F1 value(0.288) and accuracy(0.403). That means, compared with baselines, D2D-LSTM's prediction has the smallest difference from the real data.

Tree Generation Accuracy

The task of tree generation for D2D-LSTM is to generate the whole tree path with a given root node. We use tree edit distance (Paaßen 2018) to measure the difference between the ground-truth tree and the generated tree. On account of the direct proportion relationship of edit distance and the number of nodes predicted, we utilize edit distance dividing the number of nodes to balance this error.

In addition, Depth MAE is used to measure predicting the accuracy of the terminal node, and Predicting prototype Coverage for Level i (PCL_i) is used to measure predicting performance. Suppose the i-th level nodes set is D, PCL_i is defined as equation 3.

$$PCL_i = \frac{\mid \{x \in X \mid X = D_i^{ground-truth} \cap D_i^{predicted}\} \mid}{\mid \{x \in X \mid X = D_i^{ground-truth}\} \mid} \tag{3}$$

In order to measure the tree generation accuracy, we remove trees that only have two nodes in this section. The root node and its feature vector are given. We set a queue to help tree generation, and for every step, we take a node from the queue and generate its child node. Every generated child node is put in the queue. We recursively make 80 classes prediction. If a node is predicted as a terminal class, we stop child node generation for this node and take another node from the queue to continue the node generation until the queue is empty. Moreover, the child generation process will be truncated if its $childnodecount > 20$ and if its $childnodecount > 100$ to avoid an infinite error for every node's generation. Results are shown in Table 4. Our generated trees are more similar to ground truth trees than FC since we have the smallest edit distance. D2D-LSTM which is lack of location feature (D2D-LSTM@#6) has the largest MWED, which shows the importance of location feature in D2D social network. As we can see from the result, content type is not so important, because D2D transmission happens when people meet and share, which is not same as Online Social Network. Fig 9 shows a tree generated by our D2D-LSTM. Prototype 38 and prototype 37 have the same content preference and location preference, which shows the homophily of D2D transmission participants. Even though prototype 38 and prototype 26 have different content preference, prototype 38 still transfer the APP file to prototype 26. This is because of their common location preferences. So it turns out that location plays a decisive role in D2D communication.

Table 4: Tree Generation Accuracy

Model	MWED	MAE	MPCL
FC@#1	9.96	3.14	0.16
D2D-LSTM@#4	5.87	2.62	0.16
D2D-LSTM@#5	14.96	3.04	0.13
D2D-LSTM@#6	30.17	3.56	0.014
D2D-LSTM@#7	10.82	2.72	0.11
RNN@#8	18.07	2.98	0.04
D2D-LSTM@#3	**3.37**	**2.60**	**0.17**

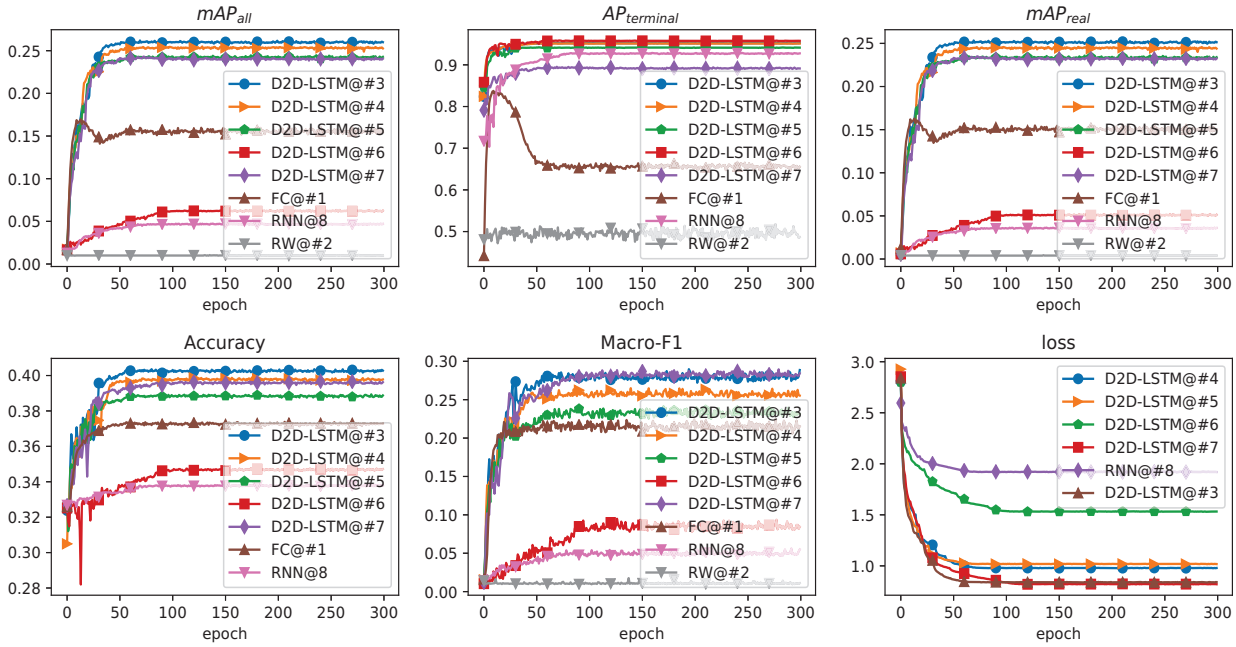

Figure 8: Performance evaluation of D2D-LSTM.

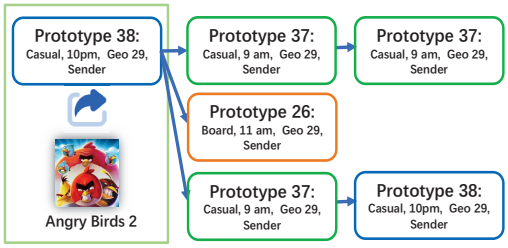

Figure 9: Tree Diffusion Path Generated By D2D-LSTM

Related Work

A large number of efforts have explored content propagation in MSN. Some previous works concentrate on influence maximization, such as (Li et al. 2017b; 2017a; 2016). Some previous works such as (Parisa et al. 2015; Li et al. 2013; Sun et al. 2014) only concentrate on single feature of the propagation process, but we aim to predict the entire content diffusion path with multi-features to increase accuracy and hit rate. (Hu et al. 2018) is mainly interested in OSN with Pinterest, but our approach is based on offline MSN. Additionally, researchers (Liben-Nowell and Kleinberg 2007; Huo, Huang, and Hu 2018; Trouillon et al. 2016; Wang et al. 2015) have made great efforts in link prediction in social network. For example, in (Liu et al. 2016), a random walk principle is put forward to calculate the possibility of one node propagating information to another node. However, link prediction has a serious drawback that it can only predict the possibility of two nodes establishing a link in a known social network topology. D2D-LSTM recursively predicts the entire diffusion paths of a content (an APP) using multi-featrues rather than predicting future edges for an existing graph. D2D-LSTM is similar to the Tree-LSTM (Chen et al. 2016b; 2016a; Tai, Socher, and Manning 2015) in natural language processing (NLP). However, Tree-LSTM in NLP is bottom-up structure, while content transmission path prediction in D2D networks must start from the root user, which makes Tree-LSTM not suitable for predicting content diffusion path in mobile social network (MSN). Zhang(Zhang, Lu, and Lapata 2015) proposed a top-down Tree-LSTM, but it requires dependency trees, trees converting from sentences in a specific order with some restrictions. Our trees are built according to users' interactions, so our trees don't meet these restrictions, causing this top-down Tree-LSTM can not be used for D2D content diffusion path prediction.

Conclusion

In this work, we present a deep recurrent network named D2D-LSTM for predicting content propagation in D2D social networks, and D2D-LSTM takes multiple dimensions into account, including the time, geography, category and transmission preferences. We derive tree propagation paths from a real-world large-scale D2D trace and create a new method converting tree path into sequence to satisfy D2D-LSTM's input format. We also utilize prototype users to increase the generalizability of the D2D-LSTM. Theoretical analysis shows that our algorithm has a low computing complexity. Empirical evaluation on the real D2D data set shows that D2D-LSTM can accurately predict the propagation path of the offline content.

References

Amer, R.; Butt, M. M.; Bennis, M.; and Marchetti, N. 2018. Inter-cluster cooperation for wireless d2d caching networks. *IEEE Transactions on Wireless Communications* 17(9):6108–6121.

Anderson, E. W. 1998. Customer satisfaction and word of mouth. *Journal of service research* 1(1):5–17.

Cha, M.; Kwak, H.; Rodriguez, P.; Ahn, Y.-Y.; and Moon, S. 2007. I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, IMC '07, 1–14. New York, NY, USA: ACM.

Cha, M.; Kwak, H.; Rodriguez, P.; Ahn, Y.; and Moon, S. 2009. Analyzing the video popularity characteristics of large-scale user generated content systems. *IEEE/ACM Transactions on Networking* 17(5):1357–1370.

Chen, Q.; Zhu, X.; Ling, Z.; Wei, S.; Jiang, H.; and Inkpen, D. 2016a. Enhanced lstm for natural language inference. *arXiv preprint arXiv:1609.06038*.

Chen, Q.; Zhu, X.; Ling, Z.; Wei, S.; and Jiang, H. 2016b. Enhancing and combining sequential and tree lstm for natural language inference. *arXiv preprint arXiv:1609.06038*.

Chevalier, J. A., and Mayzlin, D. 2006. The effect of word of mouth on sales: Online book reviews. *Journal of marketing research* 43(3):345–354.

Gers, F. A.; Schmidhuber, J.; and Cummins, F. 1999. Learning to forget: Continual prediction with lstm.

Graves, A.; Mohamed, A.-r.; and Hinton, G. 2013. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, 6645–6649. IEEE.

Gregori, M.; Gómez-Vilardebó, J.; Matamoros, J.; and Gündüz, D. 2016. Wireless content caching for small cell and d2d networks. *IEEE Journal on Selected Areas in Communications* 34(5):1222–1234.

Hao, S.; Zhang, N.; and Tao, M. 2018. A belief propagation approach for caching strategy design in d2d networks. In *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, 1–6.

Hu, W.; Singh, K. K.; Xiao, F.; Han, J.; Chuah, C.-N.; and Lee, Y. J. 2018. Who will share my image?: Predicting the content diffusion path in online social networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, 252–260. New York, NY, USA: ACM.

Huo, Z.; Huang, X.; and Hu, X. 2018. Link prediction with personalized social influence. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Li, H.; Ma, X.; Wang, F.; Liu, J.; and Xu, K. 2013. On popularity prediction of videos shared in online social networks. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, CIKM '13, 169–178. New York, NY, USA: ACM.

Li, J.; Liu, C.; Yu, J. X.; Chen, Y.; Sellis, T.; and Culpepper, J. S. 2016. Personalized influential topic search via social network summarization. *IEEE Transactions on Knowledge and Data Engineering* 28(7):1820–1834.

Li, J.; Sellis, T.; Culpepper, J. S.; He, Z.; Liu, C.; and Wang, J. 2017a. Geo-social influence spanning maximization. *IEEE Transactions on Knowledge and Data Engineering* 29(8):1653–1666.

Li, J.; Wang, X.; Deng, K.; Yang, X.; Sellis, T.; and Yu, J. X. 2017b. Most influential community search over large social networks. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, 871–882.

Liben-Nowell, D., and Kleinberg, J. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology* 58(7):1019–1031.

Liu, L.; Xu, L.; Wangy, Z.; and Chen, E. 2016. Community detection based on structure and content: A content propagation perspective. *Proceedings - IEEE International Conference on Data Mining, ICDM* 2016-Janua:271–280.

Paaßen, B. 2018. Revisiting the tree edit distance and its backtracing: A tutorial.

Parisa, T. H.; Razeghi, B.; Okati, N.; Souran, D. M.; and Mir, M. 2015. Collective wisdom based blog clustering. In *2015 6th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 1–6.

Sun, Y.; Wang, T.; Song, L.; and Han, Z. 2014. Efficient resource allocation for mobile social networks in d2d communication underlaying cellular networks. In *2014 IEEE International Conference on Communications (ICC)*, 2466–2471. IEEE.

Sundermeyer, M.; Schlüter, R.; and Ney, H. 2012. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*.

Tai, K. S.; Socher, R.; and Manning, C. D. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; and Bouchard, G. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, 2071–2080.

Wang, P.; Xu, B.; Wu, Y.; and Zhou, X. 2015. Link prediction in social networks: the state-of-the-art. *Science China Information Sciences* 58(1):1–38.

Zhang, X.; Lu, L.; and Lapata, M. 2015. Top-down tree long short-term memory networks. *arXiv preprint arXiv:1511.00060*.