Towards Explainable Networked Prediction

Liangyue Li
Arizona State University
liangyue@asu.edu

Hanghang Tong
Arizona State University
hanghang.tong@asu.edu

Huan Liu
Arizona State University
huanliu@asu.edu

ABSTRACT

Networked prediction has attracted lots of research attention in recent years. Compared with the traditional learning setting, networked prediction is even harder to understand due to its coupled, *multi-level* nature. The learning process propagates top-down through the underlying network from the macro level (the entire learning system), to meso level (learning tasks), and to micro level (individual learning examples). In the meanwhile, the networked prediction setting also offers rich context to explain the learning process through the lens of *multi-aspect*, including training examples (*e.g., what are the most influential examples*), the learning tasks (*e.g., which tasks are most important*) and the task network (*e.g., which task connections are the keys*). Thus, we propose a multi-aspect, multi-level approach to explain networked prediction. The key idea is to efficiently quantify the influence on different levels of the learning system due to the perturbation of various aspects. The proposed method offers two distinctive advantages: (1) *multi-aspect, multi-level:* it is able to explain networked prediction from multiple aspects (i.e., example-task-network) at multiple levels (i.e., macro-meso-micro); (2) *efficiency:* it has a linear complexity by efficiently evaluating the influences of changes to the networked prediction without retraining.

KEYWORDS

Explainable Networked Prediction; Influence Function

ACM Reference Format:
Liangyue Li, Hanghang Tong, and Huan Liu. 2018. Towards Explainable Networked Prediction. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3269206.3269276

1 INTRODUCTION

Networked prediction has attracted lots of research attentions in recent years. Networks, as a natural data model that captures the relationship among different objects, domains and learning components, provide powerful contextual information in modeling networked systems, including network of networks [8, 9], network of time series [2], network of learning models [4, 7]. Networked prediction has been successfully applied in many application domains, ranging from bioinformatics, environmental monitoring, infrastructure networks, to team science. By leveraging the intrinsic relationship among the networked learning components, it often brings significant performance improvement to the mining tasks, e.g., early prediction of long-term citation counts [7], network-regularized multi-task learning [4], etc.

Despite its superior prediction power, networked prediction is often hard to understand for end users, mainly due to its coupled, *multi-level* nature. At macro level, multiple learning tasks are intertwined by the task similarity network, composing the entire learning system; at meso level, a specific learning task is impacted by not only its own training examples but also other tasks; at micro level, a specific training example would potentially shape its own task and others (see Fig. 1). It is a daunting task to explain such a highly complex, multi-level learning system, which we elaborate as follows.

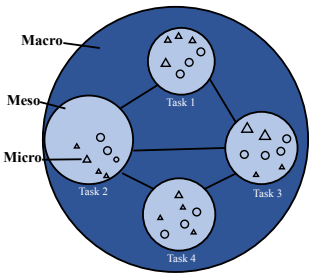

Figure 1: An illustration of networked prediction system.

At *macro* level, we want to gain a global view of how the system works, e.g., what are the ingredients that are essential to the system characteristics (e.g., the parameters of the entire system)? At the *meso* level, we focus on one specific learning task to understand its own learning behavior, e.g., how its own training samples and those from other learning tasks affect its model parameters. At the *micro* level, we focus on one specific test example and want to understand the reasons behind the prediction of this test example given by the learned models.

On the other hand, we envision that the networked prediction setting also offers rich context to explain the learning process through the lens of various aspects as follows:

- *Example aspect.* Each training example could potentially impact the learned model of the same task and that of other tasks via the underlying network. We want to identify the most influential examples at the different levels to have a comprehensive understanding of their roles in the learning process.
- *Task aspect.* A learning task, if viewed as the aggregation of its training examples, would affect the learning process of the whole system as well as each of the other learning tasks.
- *Network aspect.* A task network is essential in the networked learning system since it acts as a bridge to connect all the learning tasks together. Changing the task network would inevitably influence the learning results of the whole system.

In this paper, we propose a multi-aspect, multi-level approach to explain networked prediction. The key idea is to efficiently quantify the influence on different levels of the learning system due to the perturbation of various aspects. More concretely, the influence

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3269276

score is measured by the changes in the entire learning system's parameters (*macro*), one task's model parameters (*meso*), and the loss function value at a test sample (*micro*) in response to the changes made to the training examples, a learning task and the task network, respectively. The key advantages are (1) *multi-aspect, multi-level:* we are able to provide a comprehensive understanding to the workings of networked prediction from the perspective of multiple aspects at multiple levels, essentially through the influences of example-task-network aspects with respect to macro-meso-micro levels; and (2) *efficiency:* leveraging influence functions rooted in robust statistics [3], we can efficiently evaluate the influences of changes to the networked prediction without retraining.

2 PROBLEM DEFINITION

In this section, we present the notations used throughout the paper, and formally define the Explainable Networked Prediction problem. We use bold capital letters (e.g., $\mathbf{A}$) for matrices and bold lowercase letters (e.g., $\mathbf{w}$) for vectors.

Let us consider a networked learning system with T supervised learning tasks, for example, recognizing objects from images or predicting the sentiment from texts. The training data we have for each task is given as $\{(\mathbf{x}_i^t, y_i^t)\}_{i=1}^{n_t} \subset \mathbb{R}^d \times \mathbb{R}$, $t = 1, \ldots, T$, where n_t is the number of available training examples for the t-th task, and d is the dimensionality of the input space, which is assumed to be the same across the tasks. In this work, we consider a task relationship network described by a non-negative matrix $\mathbf{A}$ is available. In this network, each node represents a learning task and the edges represent the relatedness between the connected tasks, i.e., $\mathbf{A}_{ij}$ has a higher numerical value if the i-th and j-th tasks are closely related. The goal of networked prediction is to learn a prediction function parameterized by θ_t as $f_t(\mathbf{x}_i^t; \theta_t)$ for each task jointly in order to minimize the regularized empirical loss as follows:

$$\min_{\theta_1, \ldots, \theta_T} \sum_{t=1}^{T} \frac{1}{n_t} \sum_{i=1}^{n_t} L(f_t(\mathbf{x}_i^t; \theta_t), y_i^t) + \lambda \sum_{i=1}^{T} \sum_{j=1}^{T} \mathbf{A}_{ij} \|\theta_i - \theta_j\|^2 \quad (1)$$

where $L(\cdot, \cdot)$ is the loss function, e.g., squared loss for regression task or cross entropy loss for classification task, and the last term is to regularize the model parameters through the task relationship network $\mathbf{A}$, i.e., similar tasks share similar model parameters.

Our goal is to explain the networked learning system by quantifying the influence on different levels due to the perturbation of various aspects. More concretely, the influence score is measured by the changes in the whole learning system's parameters, one task's model parameters, and the loss function value at a test sample in response to the changes made to the training examples, a learning task and the task network.

With the above notations, the problem of explainable networked prediction can be formally defined as follows:

Problem 1. *Explainable Networked Prediction*

Given: *the training data of all the tasks* $\{(\mathbf{x}_i^t, y_i^t)\}_{i=1}^{n_t}$, *the learned models through joint training* $f_t(\cdot, \theta_t^*)$, *a query test sample from the* t*-th task* $\mathbf{x}_{\text{test}}^t$;

Compute: *the influence scores of the training samples, the learning tasks and the task network on the learning system's parameters, each task's parameters and on the prediction w.r.t.* $\mathbf{x}_{\text{test}}^t$.

3 PROPOSED MODEL

In this section, we present our model NEPAL to help explain networked prediction by measuring the influence of the various aspects (i.e., example, task, network) at multiple levels (i.e., macro/system, meso/task, micro/example). We start with a brief review of influence functions, and then present our multi-aspect, multi-level approach to explainable networked prediction, followed by analysis.

3.1 Preliminaries: Influence Function

Influence function has been used in a single learning task to efficiently evaluate the change in model parameters due to the removal of a training sample without retraining the model [5]. The key idea is to compute the parameter change should a training sample is upweighted by some small ϵ using influence function. In a nutshell, they form a quadratic approximation to the empirical loss around the model parameters and then take a single Newton step.

3.2 NEPAL – Building Blocks

In this paper, we generalize influence functions to the setting of a networked learning system, in order to evaluate the influences of multiple aspects at different levels.

A - The influence of training samples on learning system's parameters. Removing a training example from one task would not only change the parameters of the task itself, but also the parameters of other tasks through the task network. We upweight a training example $\mathbf{x}^t$ from the t-th task and compute the changes in all the tasks' model parameters. Define the new parameters of the entire learning system after such upweighting as $\theta_\epsilon^* \stackrel{\text{def}}{=} (\theta_{1,\epsilon}^*, \ldots, \theta_{T,\epsilon}^*)$ and that $\theta_\epsilon^* = \arg\min \sum_{t=1}^{T} \frac{1}{n_t} \sum_{i=1}^{n_t} L(f_t(\mathbf{x}_i^t; \theta_t), y_i^t) + \lambda \sum_{i=1}^{T} \sum_{j=1}^{T} \mathbf{A}_{ij} \|\theta_i - \theta_j\|^2 + \epsilon L(f_t(\mathbf{x}^t; \theta_t), y^t)$. The influence of the upweighting on all the tasks' model parameters can be computed as

$$\mathcal{I}_\theta(\mathbf{x}^t) \stackrel{\text{def}}{=} \frac{d\theta_\epsilon^*}{d\epsilon}\Big|_{\epsilon=0} = -\mathbf{H}_{\theta^*}^{-1} \nabla_\theta L(f_t(\mathbf{x}^t; \theta_t), y^t) \quad (2)$$

where $\mathbf{H}_{\theta^*}$ is the Hessian of the objective function defined in Eq. (1). Since removing the training example $\mathbf{x}^t$ from the t-th task is the same as upweighting it by $\epsilon = -\frac{1}{n_t}$, we can approximate the change of the parameters in the whole learning system $(\theta_{-\mathbf{x}^t}^* - \theta^*)$ by $-\frac{1}{n_t} \mathcal{I}_\theta(\mathbf{x}^t)$.

B - The influence of task network on learning system's parameters. To measure the influence of task network on model parameters, we upweight the task connection between task i and j, i.e., $\mathbf{A}_{ij}$, and use the influence function to compute the changes of the model parameters. The influence of the upweighting can be computed as

$$\mathcal{I}_\theta(\mathbf{A}_{ij}) \stackrel{\text{def}}{=} \frac{d\theta_\epsilon^*}{d\epsilon}\Big|_{\epsilon=0} = -\mathbf{H}_{\theta^*}^{-1} \nabla_\theta \mathbf{A}_{ij} \|\theta_i - \theta_j\|^2. \quad (3)$$

Since removing the connection between task i and j is equivalent to upweighting $\mathbf{A}_{ij}$ by $\epsilon = -\lambda$, we can approximate the parameter changes $(\theta_{-\mathbf{x}^t}^* - \theta^*)$ by $-\lambda \mathcal{I}_\theta(\mathbf{A}_{ij})$.

3.3 NEPAL – Proposed Approach

Based on the different aspects (i.e., training example, learning task, and task network) in the learning system, we can answer questions regarding the influences at different levels. For example, what are the most influential training samples in the whole learning system?

What are the most influential learning tasks w.r.t. a test sample? See Table 1 for a summary.

Table 1: Multi-Aspect, Multi-Level Explanation in Networked Prediction

Aspect \ Level	Macro/System	Meso/Task	Micro/Test example
Training example $\mathbf{x}^t$	globally influential training sample ($\mathcal{I}_G(\mathbf{x}^t)$)	task specific influential training sample ($\mathcal{I}_s(\mathbf{x}^t)$)	test specific influential training sample ($\mathcal{I}_{\mathbf{x}^s_{\text{test}}}(\mathbf{x}^t)$)
Learning task f_t	globally influential task ($\mathcal{I}_G(f_t)$)	task specific influential task ($\mathcal{I}_s(f_t)$)	test specific influential task ($\mathcal{I}_{\mathbf{x}^s_{\text{test}}}(f_t)$)
Task network $\mathbf{A}$	globally influential task connections ($\mathcal{I}_G(\mathbf{A}_{ij})$)	task specific influential task connections ($\mathcal{I}_s(\mathbf{A}_{ij})$)	test specific influential task connections ($\mathcal{I}_{\mathbf{x}^s_{\text{test}}}(\mathbf{A}_{ij})$)

A - Macro-level influences of training examples, tasks, and task network. We propose to use l_2-norm of the change in the whole learning system's parameters as the measure of the macro-level influence should a training sample, training samples from a task, or a task connection is removed, which can be computed as $\mathcal{I}_G(\mathbf{x}^t)=\frac{1}{n_t}\|\mathcal{I}_\theta(\mathbf{x}^t)\|_2$, $\mathcal{I}_G(f_t)=\frac{1}{n_t^2}\sum_{i=1}^{n_t}\|\mathcal{I}_\theta(\mathbf{x}_i^t)\|_2$, and $\mathcal{I}_G(\mathbf{A}_{ij})=\lambda\|\mathcal{I}_\theta(\mathbf{A}_{ij})\|_2$, respectively. Note that we use the average of the training samples' influence from a task as the influence of this learning task.

B - Meso-level influences of training examples, tasks, and task network. We propose to use the l_2-norm of the change in the parameters corresponding to one learning task as the measure of the meso-level influence should a training sample, training samples from a task, or a task connection is removed, which can be computed as $\mathcal{I}_s(\mathbf{x}^t)=\frac{1}{n_t}\|\mathcal{I}_{\theta_s}(\mathbf{x}^t)\|_2$, $\mathcal{I}_s(f_t)=\frac{1}{n_t^2}\sum_{i=1}^{n_t}\|\mathcal{I}_{\theta_s}(\mathbf{x}_i^t)\|_2$ and $\mathcal{I}_s(\mathbf{A}_{ij})=\lambda\|\mathcal{I}_{\theta_s}(\mathbf{A}_{ij})\|_2$, respectively. Note that $-\frac{1}{n_t}\mathcal{I}_{\theta_s}(\mathbf{x}^t)$ denotes the change corresponding to the parameters only in the s-th task.

C - Micro-level influences of training examples, tasks, and task network. We propose to employ the change in the loss at a particular test sample $\mathbf{x}^s_{\text{test}}$ from the s-th task as the measure of micro-level influence. We can apply chain rule to measure the influence of upweighting a training sample as

$$\mathcal{I}_\theta(\mathbf{x}^t,\mathbf{x}^s_{\text{test}}) \stackrel{\text{def}}{=} \left.\frac{dL(f_s(\mathbf{x}^s_{\text{test}};\theta^*_{s,\epsilon}),y^s_{\text{test}})}{d\epsilon}\right|_{\epsilon=0}$$
$$= -\nabla_\theta L(f_s(\mathbf{x}^s_{\text{test}};\theta^*_s),y^s_{\text{test}})^T\mathbf{H}^{-1}_{\theta^*}\nabla_\theta L(f_t(\mathbf{x}^t;\theta_t),y^t)$$

The change in loss at the test sample due to the removal of the training sample is used as the micro-level influence of $\mathbf{x}^t$ to $\mathbf{x}^s_{\text{test}}$ and can be approximated as $\mathcal{I}_{\mathbf{x}^s_{\text{test}}}(\mathbf{x}^t)=-\frac{1}{n_t}\mathcal{I}_\theta(\mathbf{x}^t,\mathbf{x}^s_{\text{test}})$. We present the algorithm for computing the micro-level influences of the training samples from all the tasks in Algorithm 1. The micro-level influences of a learning task and task network are computed as $\mathcal{I}_{\mathbf{x}^s_{\text{test}}}(f_t)=-\frac{1}{n_t^2}\sum_{i=1}^{n_t}\mathcal{I}_\theta(\mathbf{x}_i^t,\mathbf{x}^s_{\text{test}})$ and $\mathcal{I}_{\mathbf{x}^s_{\text{test}}}(\mathbf{A}_{ij})=-\lambda\mathcal{I}_\theta(\mathbf{A}_{ij},\mathbf{x}^s_{\text{test}})$, respectively. Note that $\mathcal{I}_\theta(\mathbf{A}_{ij},\mathbf{x}^s_{\text{test}})=-\nabla_\theta L(f_s(\mathbf{x}^s_{\text{test}};\theta^*_s),y^s_{\text{test}})^T\mathbf{H}^{-1}_{\theta^*}\nabla_\theta\mathbf{A}_{ij}\|\theta_i-\theta_j\|^2$.

3.4 Proofs and Analysis

In this subsection, we analyze the proposed NEPAL algorithm by giving the complexity analysis, the derivation of the key equations. We omit the proofs for brevity.

Theorem 3.1. *(*Time complexity of NEPAL). Algorithm 1 takes $O(nT^2d^2)$ with logistic regression model for each task, where $n=\sum_{t=1}^T n_t$ is the total number of training samples in all tasks and T is the number of tasks.

Algorithm 1 NEPAL - Networked Prediction Explanation

Input: (1) the training data of all the tasks $\{(\mathbf{x}_i^t,y_i^t)\}_{i=1}^{n_t}$,
(2) the learned models through joint training $f_t(\cdot,\theta_t^*)$,
(3) a query test sample from the s-th task $\mathbf{x}^s_{\text{test}}$.

Output: the micro-level influences of the training samples of all the tasks on the prediction w.r.t. $\mathbf{x}^s_{\text{test}}$.

1: Compute gradient of the loss at the test sample w.r.t. model parameters: $\mathbf{v}\leftarrow\frac{\partial L(f_s(\mathbf{x}^s_{\text{test}};\theta^*_s),y^s_{\text{test}})}{\partial\theta}$
2: Compute $\mathbf{x}=\mathbf{H}_\theta^{-1}\mathbf{v}$ by solving $\min_{\mathbf{x}}\frac{1}{2}\mathbf{x}^T\mathbf{H}_\theta\mathbf{x}-\mathbf{v}^T\mathbf{x}$ using conjugate gradient method, where the Hessian-vector product can be exactly computed using the $\mathcal{R}_{\mathbf{v}}\{\cdot\}$ operator [10]
3: **for** each task t in all tasks **do**
4: **for** $i=1,\ldots,n_t$ **do**
5: Compute the gradient of the objective function at training sample $\mathbf{x}_i^t$ w.r.t. model parameters: $\mathbf{u}\leftarrow\frac{\partial L(f_t(\mathbf{x}_i^t;\theta_t^*),y_i^t)}{\partial\theta}$
6: Compute the influence score of $\mathbf{x}_i^t$ as $\mathcal{I}_{\mathbf{x}^s_{\text{test}}}(\mathbf{x}_i^t)=\frac{1}{n_t}\mathbf{u}^T\mathbf{x}$

Lemma 3.2. *(*Correctness of Eq (2)). Denote $\mathcal{J}(\theta)=\sum_{t=1}^T\frac{1}{n_t}\sum_{i=1}^{n_t}L(f_t(\mathbf{x}_i^t;\theta_t),y_i^t)+\lambda\sum_{i=1}^T\sum_{j=1}^T A_{ij}\|\theta_i-\theta_j\|^2$, where $\theta\stackrel{\text{def}}{=}(\theta_1,\ldots,\theta_T)$. Assuming $\mathcal{J}(\theta)$ to be twice-differentiable and strictly convex, the influence of upweighting training sample $\mathbf{x}^t$ on the parameters θ can be computed by $\mathcal{I}_\theta(\mathbf{x}^t)$.

Remarks: we have a similar correctness lemma for Eq (3), which is omitted here for space. In practice, the objective function does not have to be convex. For non-convex cases, we can form a convex quadratic approximation of the loss function around the learned parameters, based on which the influence function can be computed.

4 EMPIRICAL EVALUATIONS

A - Datasets. The real world datasets used are as follows:

MNIST. MNIST [6] is a commonly used handwritten digit dataset, containing images of handwritten numerals (0-9) represented by 28×28 pixels in grayscale. We construct the networked prediction system using logistic regression for three tasks, where task 1 distinguishes digit 1 from 7, task 2 differentiates digit 2 from 7 and task 3 classifies digit 6 from 9. In the task network, we connect task 1 with 2 via $\mathbf{A}_{12}=1$ and connect task 2 with 3 via $\mathbf{A}_{23}=0.1$.

Sentiment [1]. This sentiment dataset contains product reviews from Amazon.com for many product types [1]. We build networked prediction models for reviews from *music*, *video*, *DVD*, *book* and *magazine* and a review is labeled as positive if its rating is greater than 3 and negative if below 3. We extract both unigram and bigram features from the review text. The task network is constructed based on the relevance between different product domains.

B - Results on MNIST. There are 9 experimental scenarios corresponding to the 9 cells in Table 1. Due to space limit, we only present some sampled results.

Macro-level Influences. We compute macro-level influences $\mathcal{I}_G(\mathbf{x}^t)$ for training examples of all three learning tasks.Our observation is that in all the tasks, vast majority of the examples have no or negligible influences on the entire learning system and only a few can exert significant influence. The top-*10* globally influential training examples measured by $\mathcal{I}_G(\mathbf{x}^t)$ is shown in Fig. 2 with 7 of them from second task. The top-*2* examples are the same images of digit 7 from task 2 and 1, respectively. To see how the globally influential

[1] http://www.cs.jhu.edu/ mdredze/datasets/sentiment/

examples affect the learning system's prediction performance, we flip labels of the most influential examples at macro-level, retrain the model and compute the classification accuracy on the test set. We also test the random picking strategy with 30 repetitions. Fig. 3 shows that flipping the labels of the most influential examples exert larger interruption to the learning system as demonstrated by the significant drop in the test accuracy for all the three tasks.

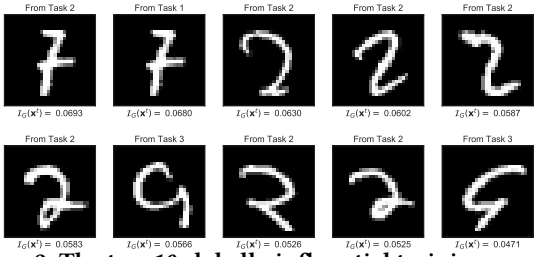

Figure 2: The top-*10* globally influential training samples.

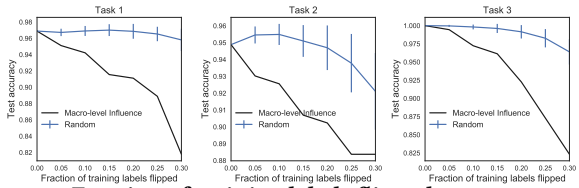

Figure 3: Fraction of training labels flipped vs. test accuracy.

Meso-level Influences. We compute meso-level influences of each learning task and observe that generally the most influential task for one specific task is the task itself, except that for the first task, task 2 has about the same influence on it as the task itself possibly due to same negative training examples of digit 7 they share. For task 2, the first task has about half influence as the task itself.

Micro-level Influences. We randomly select one test example from each of the three learning tasks and compute the micro-level influences of training samples using Algorithm 1. Fig. 4 shows the running time of Algorithm 1 with varying size of the total number of training examples n. We can see that the proposed algorithm scales linearly, which is consistent with Theorem 3.1.

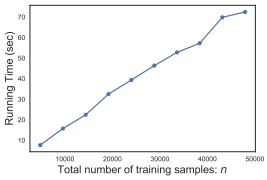

Figure 4: Running time vs. total number of training examples n.

In addition, we also compute micro-level influences of the task network specific to each test example. For the test example from the first task, $\mathbf{A}_{12}$ has a much larger negative impact than $\mathbf{A}_{23}$ possibly because connecting to task 2 does not help with the prediction of digit 1. For the test example of digit 7 from task 2, $\mathbf{A}_{12}$ has a larger positive impact than $\mathbf{A}_{23}$ since connecting to task 1 can help with this prediction.

C – Case Studies on Sentiment Dataset. We use the test example from *music* category and show the most influential training examples from other domains in Table 2. Comparing the test example and the helpful training examples (i.e., those with large positive influence scores) from all the domains, it seems they are overall towards the positive sentiment despite some negative descriptions about the products, e.g., the book "seems really dull" to average readers in the *book* domain. The harmful example (i.e., those with large negative influence scores) from *music* is labeled as negative sentiment but the descriptions still sound largely positive.

Table 2: Case study on *Sentiment*.

	Review Text (positive sentiment is highlighted in bold font and negative sentiment is highlighted with underline). [...] is used to omit some sentences without altering the main meaning of the text.	**Label**
Test example from *music*	I was **instantly drawn into her music**. What I love about her songs is that they are so real. "You Give Me Love" is **so real and so strong**. "The Secret of Life" has taken some getting used to. It is not my absolute favorite on the CD. "Me" is the song that **means the most to me** since I have experienced trying to change for someone else in a relationship. [...] She has a big voice, and she **nails each song** on this CD. Give it a try.	+
Harmful example from *music*	**I liked this** when it first came out b/c I was 16.[...]This is **their best work** since the weirdness does get lame after a while. Favorite song is Mongoloid...totally strange **but rocking song**. "Uncontrollable Urge" **rocks** (in a weird way). Satisfaction is completely unique but its not a good song. [...] I saw them live (they were horrible) during their hey day. [...]**Anybody that gives this novelty group 5 stars is cheapening what true excellent music is.**	−
Helpful example from *book*	Here's a good litmus test to show **how good a book like "Breathing Lessons" is**–nothing extraordinary happens and yet I did not want to put the book down. [...]To the average reader this book probably would seem really dull. Heck, if someone told me the plot of this book I'd think it was really dull too, but **I didn't want to put it down**. [...] **It's hard for me to find any real faults with this book**, except for the lengthy flashback near the end that perhaps goes on too long. Some people may call this boring or dull, but I would call it purely exceptional. **I LOVED this book and highly recommend it**	+

5 CONCLUSION

In this paper, we propose a multi-aspect, multi-level approach NEPAL to explain networked prediction by characterizing how the learning process is diffused at different levels from different aspects. The key idea is to efficiently quantify the influence on different levels (i.e., macro/system, meso/task, micro/example) of the learning system due to the perturbation of the various aspects (i.e., example, task, network). The empirical evaluations on real-world datasets demonstrate the efficacy of the proposed NEPAL algorithm.

ACKNOWLEDGMENTS

This work is supported by National Science Foundation under Grant No. IIS-1651203, IIS-1715385, CNS-1629888 and IIS-1743040, DTRA under grant number HDTRA1-16-0017, by Army Research Office under contract number W911NF-16-1-0168, Department of Homeland Security under Grant Award Number 2017-ST-061-QA0001.

REFERENCES

[1] John Blitzer, Mark Dredze, Fernando Pereira, et al. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, Vol. 7. 440–447.
[2] Yongjie Cai, Hanghang Tong, Wei Fan, and Ping Ji. 2015. Fast Mining of a Network of Coevolving Time Series. In *SDM*. 298–306.
[3] R Dennis Cook and Sanford Weisberg. 1982. *Residuals and influence in regression*. New York: Chapman and Hall.
[4] Tsuyoshi Kato, Hisashi Kashima, Masashi Sugiyama, and Kiyoshi Asai. 2008. Multi-task learning via conic programming. In *NIPS*. 737–744.
[5] Pang Wei Koh and Percy Liang. 2017. Understanding Black-box Predictions via Influence Functions. In *ICML*. 1885–1894.
[6] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998).
[7] Liangyue Li and Hanghang Tong. 2015. The Child is Father of the Man: Foresee the Success at the Early Stage. In *KDD*. 655–664.
[8] Jingchao Ni, Hanghang Tong, Wei Fan, and Xiang Zhang. 2014. Inside the atoms: ranking on a network of networks. In *KDD*. 1356–1365.
[9] Jingchao Ni, Hanghang Tong, Wei Fan, and Xiang Zhang. 2015. Flexible and Robust Multi-Network Clustering. In *KDD*. 835–844.
[10] Barak A Pearlmutter. 1994. Fast exact multiplication by the Hessian. *Neural computation* 6, 1 (1994), 147–160.