Time-sensitive and Evolution-based Entity Resolution in Heterogeneous Information Spaces

Dan Yang, Mo Chen

Abstract-Entities are heterogeneous and usually with temporal information in heterogeneous information spaces, and their attribute values and associated entities evolve over time. This paper proposes a time-sensitive and evolution-based entity resolution (ER) clustering algorithm, TSE-Clustering which considers the evolution effect and temporal information of entity. TSE-Clustering adopts evolution-based attribute and relational similarity, and temporal trend similarity when computing reference similarity. Moreover when computing relational similarity, it considers interaction effect of evolutions by introducing the concepts of evolution pair and evolution chain. It leverages time-sensitive constraints besides semantic constraints during clustering process. Experimental results show the effectiveness and scalability of the proposed ER approach.

Index Terms—entity resolution, heterogeneous information spaces, time-sensitive, evolution

I. INTRODUCTION

Entity Resolution (ER) is the process of identifying and merging representations judged to represent the same real-world entity. In heterogeneous information spaces, entities usually contain references over a long period of time, and the attribute values of each reference are usually associated with timestamps or timespans to describe some facts of a real-world entity at that particular time. So we often need to identify the references with temporal information that describe the same entity over time which enables data integration and interesting data analysis over time.

A. Motivation

One of the important features of heterogeneous information spaces is evolution. The entities and their associated entities in them often evolve over time. In order to illustrate challenges of ER in heterogeneous information spaces, let us consider the following toy example of evolution of an author entity and its associated entities from bibliographic domain shown in Fig. 1. From Fig. 1 we can observe that:

Observation 1: When the author's affiliation changed, his/her email, supervisor also changed accordingly. One attribute value evolving may cause evolutions of other attribute values. This can be seen as evolution within entities.

Observation 2: The author's associated entities (e.g., papers, projects, and conferences) also evolved over time. Evolution of one entity may cause other associated entities to evolve directly or indirectly. This can be seen as global evolution among entities. Moreover, the evolution speeds of different related entities may not be synchronized due to different connecting strengths with the entity or different response time to the evolution.

Another example is from healthcare domain which includes associated entities of doctors, patients, nurses, diseases, drugs, treatments, hospitals, insurance providers and medical equipment etc. When a patient's disease evolves from liver cirrhosis to liver cancer over time, the doctor and hospital the patient choosing are also changing accordingly.

Moreover, the two observations above often happen simultaneously which causes more intricacies to ER. If blindly ignoring the evolution effects, we are likely to get false ER results.

B. Challenges and Contributions

Although many research efforts have been conducted in ER, ER in heterogeneous information spaces still faces many challenges, including but not limited to: (1) how to capture the evolution features of entities and to leverage useful temporal information of entity effectively. (2) how to measure evolution effects on entity simliarity correctly. (3)

Manuscript received March 9, 2017; revised May 16, 2017. This work was supported by the National Natural Science Foundation of China (No. 61402213，61402093).

Dan Yang is with School of Software, University of Science and Technology LiaoNing, Anshan, China (phone: 86-412-5929818; fax: 86-412-5929818; e-mail: asyangdan@ 163.com).

Mo Chen is with School of Computer Science and Engineering Northeastern University, Shenyang, China (e-mail:, chenmo@mail.neu.edu.cn).

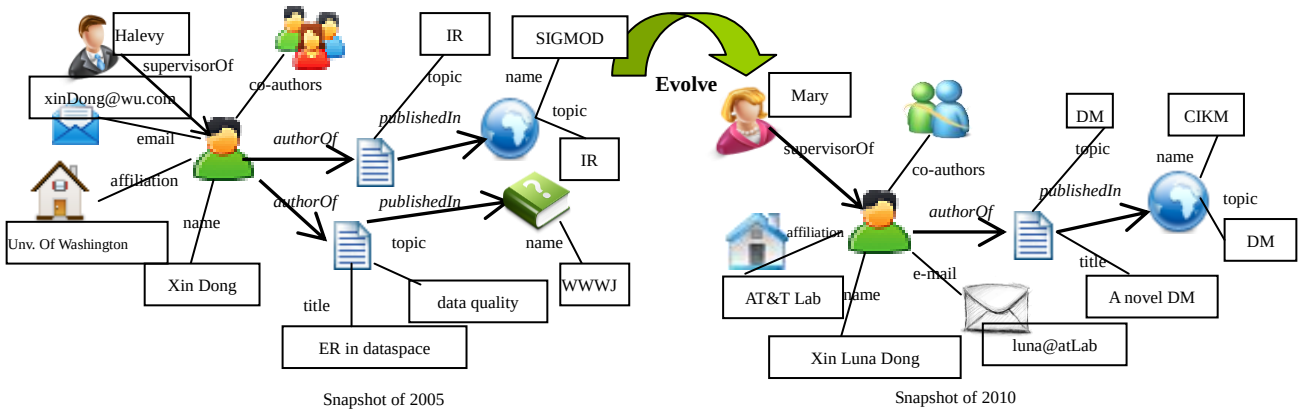


Fig.1. Toy example of evolution of an author entity and its associated entities

how to apply time-sensitive constraints effectively to ER algorithm besides traditional semantic constraints. Currently existing ER algorithms mostly assume that entity and its associated entities are time invariant, without considering the evolution features of entities, ignore or not fully leverage entity's temporal information. So they cannot be well applied to ER in heterogeneous information spaces.

To address the above challenges, we propose a time-sensitive and evolution-based ER algorithm TSE-Clustering which considers evolution effect and fully leverages temporal information of entity. The main contributions of this paper are three-fold: (1) We formulate the problem of time-sensitive and evolution-based ER in heterogeneous information spaces and propose TSE-Clustering algorithm. (2) We fully consider the temporal information and evolution effect of entity, and propose time-sensitive and evolution-based similarity measures by introducing attribute evolution coefficient and relation evolution coefficient, and temporal trend similarity. Moreover we leverage time-sensitive constraints besides semantic constraints in the clustering algorithm. (3) We have done extensive experiments on real and synthetic data respectively to evaluate effectiveness of our algorithm.

II. Related Work

ER is not a new research problem which has attracted much attentions and research efforts of different fields. The existing ER related works are mainly on relational database (also called record linkage [1]), Web [2], complex data [3] (e.g., XML data, graph data and complex network) and personal dataspace [4] etc. And it has many special applications such as product matching [5], business chain identification (linking spatial records) [6], matching offers to products [7]. With respect to entity similarity measures, early and traditional technique is featured-based (Attribute-based) ER [8]. The shortcoming of it is not accurate and too simple in some circumstances. So relational ER technique [9-11] is proposed which also considers relations among entities (such as *co_authorOf*, *citationOf*) besides feature similarity when computing similarity. Focus on improving ER accuracy more recent approaches consider the collective ER or Joint ER [12-14] to improve accuracy of resolution results.

However two important considerations are missing from the above works: (1) the temporal information of entities; (2) the evolution features of entities and their relations. Entity evolution makes ER more complex as it introduces the temporal dimension. In the pioneering work, temporal clustering algorithm [15] studies the problem of temporal records linkage considering evolution of entities. It proposes building a temporal model "time decay model" that learns how entities evolve in a labeled data set. Related work [16] follows the approach in [15] to learn temporal model and proposes two-phase clustering algorithm SFDS to matching temporal records which achieves equivalent matching accuracy but takes far less time. Related work [17] presents "mutation model" to model entity evolution for temporal record matching which focuses on the probability of a value re-appearing over time and only considers evolution within entities. And related work [18] proposes transition model to capture the probability of changing attribute values over time. Our work differs from [15-18] in the following ways: (1) TSE-Clustering considers and leverages heterogeneous entity relations and their associations. (2) We propose a more

sophisticated temporal model which also considers inter-entity evolution besides intra-entity evolution in [15, 17]. In fact, evolution does not hold for entities only but also for their associated entities. Though [15-18] have all considered entity evolution effect, they do not consider the complex evolution chain reaction pattern on different associated entities. While our approach introduces concepts of evolution pair, evolution chain and evolution damping factor to capture the implicit evolution correlation of associated entities. (3) We propose a more reasonable reference similarity metrics leveraging temporal trend analysis. [15] is based on attribute similarity when computing record similarity, while TSE-Clustering fully uses the temporal evidences by adopting similarity measures combining evolution-based attribute similarity, evolution-based relational similarity and temporal trend similarity.

III. Problem formulation and TSE-Clustering Overview

A. Preliminary

According to attribute value is invariant over time or not, we classify entity attributes into two categories: (1) time variant attribute (*tva*), e.g., people's name, email address, telephone number, affiliation, address; (2) time invariant attribute (*tia*）, e.g., person's birthday, native place, graduate year. The set of *tva*s and *tia*s of an entity class are denoted as TVA and TIA respectively.

Similarly, according to whether a relation is invariant over time or not, we classify relations among entities into two categories: (1) time variant relation (*tvr*); (2) time invariant relation (*tir*）. E.g., *co-authorOf*, *subordinateOf*, *advisorOf* relations are all *tvr*s, while *parentOf*, *grandfatherOf* relations are *tir*s. The set of *tvr*s and *tir*s of an entity class are denoted as *TVR* and *TIR* respectively.

To note here, time invariant attribute can be seen as a special case of time variant attribute. Similarly time invariant relation is a special case of time variant relation.

From the two observations we know that, evolution of one attribute can affect another attribute or associated entities directly or indirectly. So we introduce the concept of evolution pair.

DEFINITION 1. (Evolution Pair denoted as *ep*) is a pair to indicate when one *tva* or *tvr* evolves over time, another *tva* value or associated entities on *tvr* may change. The format of an evolution pair is *ep* :<*tva*/*tvr*, *tva*/*tvr*> , where $tva \in TVA$; $tvr \in TVR$.

Evolution pair indicates a dependent evolving relationship. E.g., *ep*: <*affiliation*, *co_authorOf*> indicates a person's affiliation changes over time, the associated entities on co_authorOf relation probably change accordingly. *ep*: <*affiliation*, *subordinateOf*> indicates person's affiliation evolved over time, associated entities on relation '*suborinateOf*' maybe also change with it. *ep*s of an entity class can be defined by domain experts, or be mined from the data.

Evolution pair can be further divided into heterogeneous evolution pair and homogeneous evolution pair.

Homogeneous evolution pair (*Homo ep*). Attribute evolution of one entity class can directly affect other attributes of the same entity class. E.g., the eps:<Author's affiliation, Author's research interest>, <Author's affiliation, Author's supervisor>.

Heterogeneous evolution pair (*Heter ep*). Attribute evolution of one entity class can implicitly affect attributes of other entity classes due to direct or indirect associations (instances of relations) among heterogeneous entities. E.g., in Fig. 1, there are associations among three heterogeneous entity classes, $Author \xrightarrow{authorOf} Paper \xrightarrow{publishedIn} Conference$.

So there may be the following *ep*s: <*Author's affiliation*, *Paper's topic*>, <*Author's affiliation*, *Conference's topic*>, <*Paper's topic*, *Conference's topic* >, <*Author's supervisor*, *Paper's topic*>.

Properties of evolution pair: (1) **Transitivity**. $\forall ep_1 < tvr_1, tvr_2 >, ep_2 < tvr_2, tvr_3 > \in EP, ep_1 \wedge ep_2 \Rightarrow ep_3 < tvr_1, tvr_3 >$. E.g., if there are both *ep*:<*Author's affiliation*, *supervisorOf*> and *ep*: <*supervisorOf*, *Author's research interest*>, then we can infer *ep*: <*Author's affiliation*, *Author's research interest*>. (2) **Asymmetry**. $ep_1 : < tva_1 / tvr_1, tva_2 / tvr_2 > \Rightarrow \neg ep_2 : < tva_2 / tvr_2, tva_1 / tvr_1 >$ E.g., if there is *ep*: <*Author's affiliation*, *co-authorOf*>, but intuitionally we cannot infer *ep*:< *co-authorOf*, *Author's affiliation*>.

According to the properties of evolution pair, we can know

that evolution of an entity can cause evolution chain reaction of its associated entities. So we introduce the concept of evolution chain.

DEFINITION 2. (Evolution Chain denoted as *e-Chain*) is chain composed of more than three *tva*/*tvr* items to indicate that the first item evolving over time causes other items to change gradually along the chain. The format of an *e-Chain* is: $tva_1 \prec tva_2/tvr_2 ... \prec tva_n/tvr_n$ which represents an evolutionary sequence of time variant attributes and time variant relations.

The evolution damping factors (denoted as *df*) of *e-Chain*: Intuitively, some change maybe delay and non-real-time, that is to say the evolution effect gradually decreases along the *e-Chain*. E.g., for the following *e-Chain*: $Author's\ affiliation \prec supervisorOf \prec Author's\ research\ interest \prec Paper's\ topic \prec ConferenceOf$, assume the dfs associated with each *tvr* (denoted as df_{tvr}) along the e-Chain are 0.1 0.4, 0.6 and 0.8 respectively, then we have Author's $affiliation \prec_{0.1} supervisorOf \prec_{0.4} Author's\ research\ interest \prec_{0.6} Paper's\ topic \prec_{0.8} ConferenceOf$. $df \in [0,1]$, is increasing along the *e-Chain*, namely $\forall tvr \in e\text{-}Chain$, $tvr_1 \prec tvr_2$, $df_{tvr1} \le df_{tvr2}$. *df*s of an *e-Chain* can be defined by domain experts, or be learnt from the data.

Intuitionally associated entities of an entity evolve over time, but different entities do not evolve at the same speed. So along the *e-Chain* low similarities on some relations does not indicate un-match. Intuitively speaking, the *dfs* tell us how to weight relational similarities on various relations.

Example 1. Assume the relational similarity of two author's Supervisors is 0, and the similarity of their Conferences is 1. Suppose the weights of Supervisor and Conference are both 0.5. So the $Sim_{Rel}(r_1,r_2)$= 0.5*0+0.5*1/(0.5+0.5) =0.5. After added the *df*: the similarity is $Sim_{Rel}(r_1,r_2)$=0.2*0.5*0+0.8*0.5*1/(0.2*0.5+0.8*0.5)=0.8. By applying *df* we can merge the two references.

B. Problem Specification

We assume that entity references in the heterogeneous information spaces are all with temporal information (e.g., timestamp *t*, timespan *ts*) correlated with their attribute values or associations. E.g., a paper published in 2007 is denoted by "ER method overview@2007", a *supervisorOf* association during 2003 and 2007 is described as: $\text{Halevy} \xrightarrow{supervisorOf@[2003,2007]} \text{Xin Luna Dong}$.

DEFINITION 3. (Time-sensitive and evolution-based ER in heterogeneous information spaces) is the process of identifying references with temporal information of different entity classes ($\{ec_1,ec_2,\ldots,ec_n\}$ $n \ge 1$) judged to represent the same real-world entity which considers evolution effect and temporal information of entity. The input is set of references with temporal information $R=\{r_1,r_2,\ldots,r_{|R|}\}$ of an entity class ec_i, where r_i consists of a set of attributes $A=\{attribute_1, attribute_2,\ldots, attribute_{|A|}\}$ with corresponding attribute values $V=\{(value_1,@\{t \text{ or } ts\}), (value_2, @\{t \text{ or } ts\}),\ldots,(value_{|A|}, @\{t \text{ or } ts\})\}$; *t*, *ts* represent timestamp, timespan correlated with each attribute value respectively; the output is set of clusters $Cec=\{c_1, c_2,\ldots, c_{|C|}\}$, and each cluster ci represents one real-world entity.

The resolution result is that references in the same cluster refer to the same real-world entity and different clusters refer to different entities.

C. Overview of TSE-Clustering

We treat ER in heterogeneous information spaces as an unsupervised clustering problem. Our clustering algorithm leverage temporal information of references in two ways. First, we propose time-sensitive and evolution-based similarity measures considering temporal trend similarity and evolution effects when computing reference similarity. Second, when merging two clusters according to cluster similarity, besides traditional semantic constraints, we also consider time-sensitive constraints to decide whether to merge two clusters.

Our goal is to develop an ER algorithm fully leveraging the useful temporal information and evidences of entity to boost ER performance in heterogeneous information spaces.

IV. TIME-SENSITIVE AND EVOLUTION-BASED SIMILARITY MEASURES

In this section we first introduce the similarity measures used in TSE-Clustering algorithm; then introduce evolution coefficient (*aec*), relation evolution coefficient (*rec*) and

evolution damping factor (*df*) learning methods. Finally we present the temporal trend similarity in detail. To ease the discussion that follows, we summarize the notation we use in Table I.

TABLE I. SUMMARY OF NOTATION

Notation	Descriptions
tva	a time variant attribute of an entity class
TVA	set of time variant attributes of an entity class
tia	a time invariant attribute of an entity class
TIA	set of time invariant attributes of an entity class
tvr	a time variant relation of an entity class
TVR	set of time variant relations of an entity class
tir	time invariant relation of an entity class
TIR	set of time invariant relations of an entity class
aec	attribute evolution coefficient
rec	relation evolution coefficient
ep	evolution pair
e-Chain	evolution chain
df	evolution damping factor

A. Similarity Measures of TSE-Clustering

Similarity computation of TSE-Clustering take advantage of a complete view of entity and their associated entities evolution. The similarity of two clusters c_i, c_j denoted as $SIM(c_i, c_j)$ (see formula (1)) is composed of evolution-based attribute similarity denoted as $Sim_{TA}(c_i, c_j)$, evolution-based relational similarity denoted as $Sim_{TRel}(c_i, c_j)$ and temporal trend similarity denoted as $Sim_{Ttrend}(c_i,c_j)$.

$$SIM(c_i,c_j)=\beta\cdot(\alpha\cdot Sim_{TA}(c_i,c_j)+(1-\alpha)\cdot Sim_{TRel}(c_i,c_j))+(1-\beta)\cdot Sim_{Ttrend}(c_i,c_j) \quad (1)$$

Where α,1-α are the weights of attribute similarity and relational similarity respectively and $\alpha\in[0,1]$; 1-β is weight of temporal trend similarity and $\beta\in[0,1]$. When α=1 and β=1, formula (1) becomes traditional attribute/feature based similarity measure. When each cluster includes multiple references, i.e., $|c_i|>1$, $|c_j|>1$, $Sim_{TA}(c_i,c_j)$ needs computing aggregate similarity of two clusters, simple substitute method is to adopt the max reference similarities of the two clusters, namely $Sim_{TA}(c_i,c_j) \approx max(Sim_{TA}(r_i,r_j))$ $r_i\in c_i$, $r_j\in c_j$. Likewise, $Sim_{TRel}(c_i,c_j) \approx max(Sim_{TRel}(r_i,r_j))$ $r_i\in c_i$, $r_j\in c_j$, $Sim_{Ttrend}(c_i,c_j) \approx max(Sim_{Ttrend}(r_i,r_j))$ $r_i\in c_i$, $r_j\in c_j$.

1) Evolution-based Attribute Similarity

Inspiring by the existing work of learning decay in [15], we adapted from the techniques of it. Attribute evolution coefficient (*aec*) is introduced when computing attribute similarity of two references considering evolution effect of entity attributes.

Attribute evolution coefficient: there are two types of *aec*s: (1) the probability that two different entities share the same *tva* value within time Δt, denoted as $aec_{=}(tva,\Delta t)$. The purpose of $aec_{=}$ is to reduce reward for high attribute similarity of different entities over time evolution; (2) the probability that an entity changes its *tva* value within time Δt, denoted as $aec_{\neq}(tva,\Delta t)$. The purpose of $aec_{\neq}$ is to reduce penalty for low attribute similarity of same entity over time evolution. *aec* can be defined by domain expert or learnt from the labeled data. $aec\in[0,1]$, satisfies time monotonicity, namely $\forall tva\in TVA$, $\Delta t_1<\Delta t_2$, $aec(tva,\Delta t_1)\le aec(tva,\Delta t_2)$. Take author entity's attributes as example, $aec_{=}(name,\Delta t=5)=0.05$, $aec_{\neq}(affiliation,\Delta t=4)=1/(2+0)=0.5$, $aec_{\neq}(affiliation,\Delta t=5)=0.9$.

After adding *aec*, the attribute similarity of two references is shown in formula (2), one part is *TIA* similarity, and the other part is *TVA* similarity. μ,1−μ are weights of each part respectively.

$$Sim_{TA}(r_1,r_2)=\mu\cdot\sum_{j=1}^{|TIA|}w_j\cdot sim_a(r_1.tia_j,r_2.tia_j)+(1-\mu)\cdot\frac{\sum_{i=1}^{|TVA|}(1\text{-}aec(tva_i,\Delta t))\cdot sim_a(r_1.tva_i,r_2.tva_i)}{\sum_{i=1}^{|TVA|}(1-aec(tva_i,\Delta t))} \quad (2)$$

Where w_j is weight on each *tia* and $\sum w_j=1$. $\Delta t=|r_1.t\text{-}r_2.t|$, Sim_a is a traditional attribute similarity measure on tia such as edit distance, jaro-winker distance. $1\text{-}aec_i(tva_i,\Delta t)$ is weight on each *tva* and

$$1\text{-}aec(tva_i,\Delta t)=\begin{cases}1-aec_{=}(tva_i,\Delta t) & if\ sim_a>\theta_{High}\\ 1-aec_{\neq}(tva_i,\Delta t) & if\ sim_a<\theta_{Low}\end{cases} \quad (3)$$

Example 2. Assume attribute similarity (e.g., name,

affiliation) of references r_1 and r_2 are 0.9 and 0 respectively, and we use weight 0.5 for both of them, so the attribute similarity without considering evolution effect is $Sima(r_1,r_2)=\frac{0.5*0.9+0.5*0}{0.5+0.5}=0.45$. Now assume $aec_{=}(\text{name},\Delta t=5)=0.05$, $aec_{\neq}(\text{affiliation},\Delta t=5)=0.9$, after taking *aec* into consideration, the evolution-based attribute similarity of r_1 and r_2 is $Sim_{TA}(r_1,r_2)=\frac{(1-0.05)*0.9+(1-0.9)*0}{(1-0.05)+(1-0.9)}=0.81$.

2) Evolution-based Relational Similarity

When computing reference similarity, one of naïve and simple methods is to flatten data and treat entity's relations as attributes. But shortcoming of this method is that it may lose some structural information and not very accurate sometimes. So in this paper we distinguish attributes and relations of entities. Similar to entity attributes, relations may evolve over time. We introduce relation evolution coefficient (*rec*) when computing relational similarity of two references considering evolution effect of entity relations.

Relation evolution coefficient (*rec*): The purpose of rec is to reduce penalty for low relational similarity (θ_{Low}) of same entity over time evolution. It is the probability that associated entities of an entity on *tvr* changes within time Δt, denoted as $rec_{\neq}(tvr,\Delta t)$. *rec* can be defined by domain expert or learnt from the labeled data. $rec\in[0,1]$, satisfies time monotonicity, namely $\forall tvr\in TVR$, $\Delta t_1<\Delta t_2$, $rec(tvr,\Delta t_1\le rec(tvr,\Delta t_2)$. Take author entity's relations as example, $rec_{\neq}(co\_authorOf,\Delta t=1)=0.6$, $rec_{\neq}(co\_authorOf,\Delta t=3)=0.8$, $rec_{\neq}(co\_authorOf,\Delta t=5)=0.9$, $rec_{\neq}(SubordinateOf,\Delta t=1)=1$, $rec_{\neq}(SubordinateOf,\Delta t=1)=1$.

Unlike *aec*, here, we do not have to consider $rec_{=}$. Because *tvr* and *tva* differ in that: (1) Having different values for attributes of associated entities on *tvr* does not necessarily mean two references unmatched; and (2) Sharing similar associated entities is treated as additional evidence for references match.

After adding $rec_{\neq}$, the relational similarity of two references is shown in formula (4) which concludes two parts with weights $\gamma,1-\gamma$ respectively, one part is relational similarity on *TIR*, and the other part is relational similarity on *TVR*.

$$Sim_{TRel}(r_1,r_2)=\gamma\cdot\sum_{j=1}^{|TIR|}s_j\cdot Sim_{rel}(r_1.tir_j,r_2.tir_j)+(1-\gamma)\cdot\frac{\sum_{i=1}^{|TVR|}(1-rec_{\neq}(tvr_i,\Delta t))\cdot Sim_{Rel}(r_1.tvr_i,r_2.tvr_i)}{\sum_{i=1}^{|TVR|}(1-rec_{\neq}(tvr_i,\Delta t))} \quad (4)$$

Where Sim_{rel} is any traditional relational similarity; $r_1.tir_j$ and $r_2.tir_j$ are associated entities on relation tir_j of r_1 and r_2 respectively; s_j is weight on each *tir* and $\sum s_j=1$. $r_1.tvr_i$ and $r_2.tvr_i$ are associated entities on relation tvr_i of r_1 and r_2 respectively.

Example 3. Assume relational similarity of references r_1 and r_2 on *co_authorOf* and *subordinateOf* are 0.8 and 0 respectively, weights are both 0.5, so the relational similarity of r_1 and r_2 is $Sim_{Rel}(r_1,r_2)=\frac{0.5*0.8+0.5*0}{0.5+0.5}=0.4$. Now taking consideration of $rec_{\neq}$, assume $rec_{\neq}(co\_authorOf,\Delta t=1)=0.6$, $rec_{\neq}(subordinateOf,\Delta t=1)=1$, the evolution-based relational similarity of r_1 and r_2 is $Sim_{Rel}(r_1,r_2)=\frac{(1-0.6)*0.8+(1-1)*0}{(1-0.6)+(1-1)}=0.8$.

B. Learning aec , rec and df

1) Learning *aec*

Three entities e_1, e_2 and e_3, their *tva* values evolution is shown in Fig.2. For each e_i, the points t_i ($i\ge1$) on the timeline denote different timepoints, v_{ij} indicates the j_{th} different attribute value of *tva*, its timestamp is $v_{ij}.t$; Δt_{ij} is j_{th} timespan. ΔT_{mn} denotes timespan between two timepoints on that entities e_m and e_n share the same value v_{same}, i.e., $\Delta T_{mn}=|e_m.v_{same}.t-e_n.v_{same}.t|$. On each entity's timeline value of last time point is same with that of the previous time point. If entity e_i only has one time point on its timeline, then the point is not only value change point, also last time point. The $aec_{\neq}$ is calculated as follows:

$$aec_{\neq}(tva,\Delta t)=\frac{|\{l\in L_f\mid l\le\Delta t\}|}{|L_f|+|\{l\in L_p\mid l\ge\Delta t\}|}. \quad (5)$$

Where Lp denotes set of timespans, each of which is composed of last time point and its previous time point; L_f denotes set of timespans composed of each two value change points. E.g., in Fig. 2, $Lp=\{\Delta t_{11},\Delta t_{22},\Delta t_{32}\}$ and $L_f=\{\Delta t_{21},\Delta t_{31}\}$. Take e_2 as an example, it has two timespans, i.e., Δt_{21} and Δt_{22}, $\Delta t_{21}\in L_f$ and $\Delta t_{22}\in L_p$.

Suppose $\Delta t_{21}=4$, $\Delta t_{31}=3$, $\Delta t_{11}=5$, $\Delta t_{22}=2$, $\Delta t_{32}=6$, then $aec_{\neq}(tva,\Delta t=3)=1/(2+2)=0.25$,

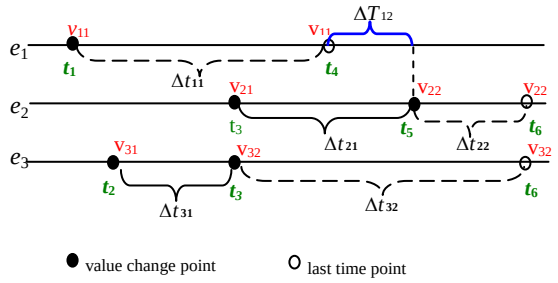


Fig. 2. Example of learning *aec*

$aec_{\neq}(tva,\Delta t=4)=2/(2+2)=0$.

Similarly, the $aec_{=}$ is calculated as follows:

$$aec_{=}(tva,\ \Delta t)=\frac{|\{l\in L_{span}\,|\,l\le\Delta t\}|}{|L_{span}|}. \quad (6)$$

Where L_{span} denotes set of timespans of any two different entities with the same attribute value. E.g., in Fig. 2, e_1 at timespan $[t_1,t_4)$, e_2 at timespan $[t_5,t_6)$ share the same attribute value, i.e.,v_{12}=v_{22},then $\Delta T_{12}=t_5-t_4$. While e_1 and e_3, e_2 and e_3 have no same attribute value on the timeline, we use signal '$\perp$' to describe it, then $L_{span}=\{\Delta T_{12},\perp,\perp\}$. Assume $\Delta T_{12}=3$, so we have $aec_{=}(tva,\ \Delta t=3)=0/3=0$, $aec_{=}(tva,\ \Delta t\ge 3)=1/3=0.33$.

2) Learning *rec*

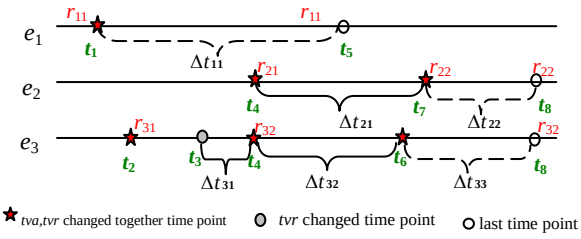


Fig.3. Example of learning *rec*

Make a little change on Fig. 2, and assume <*tva*, *tvr*> is an evolution pair of entity class of entities e_1, e_2 and e_3, we add associated entity on *tvr* to each time point $v_{ij}.t$ on e_i's timeline. So evolution of e_1, e_2 and e_3 on *tvr* along the timeline is shown in Fig. 3. For each entity e_i, r_{ij} represents the j_{th} different associated entity. Similar to learning aec, $rec_{\neq}$ is calculated as

$$rec_{\neq}(tvr,\ \Delta t)=df_{tvr}\cdot\frac{|\{l\in L_f\,|\,l\le\Delta t\}|}{|L_f|+|\{l\in L_p\,|\,l\ge\Delta t\}|}. \quad (7)$$

Where df_{tvr} is evolution damping factor of *tvr* in *e-Chain* accordingly; L_p denotes set of timespans, each of which is composed of *tva*, *tvr* changed together time point and the last time point of an entity; L_f denotes set of timespans composed of each two *tva*, *tvr* changed together time points. E.g., in Fig.3, $L_p=\{\Delta t_{11},\ \Delta t_{22},\ \Delta t_{33}\}$, $L_f=\{\Delta t_{21},\ \Delta t_{31},\ \Delta t_{32}\}$. Take e_3 as an example, it has three timespans, i.e., Δt_{33}, Δt_{31} and Δt_{32}; where $\Delta t_{33}\in L_p$, $\{\Delta t_{31},\ \Delta t_{32}\}\in L_f$. Suppose $\Delta t_{11}=5$, $\Delta t_{22}=2$, $\Delta t_{32}=4$, $\Delta t_{21}=4$, $\Delta t_{31}=1$, $\Delta t_{32}=3$, $\Delta t_{32}=3$, then $rec_{\neq}(tvr,\Delta t=3)=2/(3+2)=0.4$, $rec_{\neq}(tvr,\Delta t=4)=3/(3+2)=0.6$.

3) Learning *df*

The *e-Chain*'s ($tva_1\prec tvr_1\prec tvr_2\prec tvr_3$) timeline of entity *e* is shown in Fig. 4. The *df* associated with each *tvr* in the e-Chain is calculated as follows:

$$df_{tvr}=\sum_{e\in E}\frac{|T-\Delta t_{tvr}|}{|T|}\Big/|E|. \quad (8)$$

where T denotes the whole time of *e-Chain* spans of e. Δt_{tvr_i} denotes the elapsed time after tva_1 value changed when associated entity on tvr_i changing, i.e., $\Delta t_{tvr_i}=[tva_1,trv_i]$. *E* is entity set, |*E*| is the number of entities in E. E.g., in Fig. 4, suppose |*E*|=1, i.e., *E*={*e*}, $\Delta t_{tvr1}=1$, $\Delta t_{tvr2}=4$ and *T*=5, then df_{tvr1}=(5-1)/5=0.8, df_{tvr2}=(5-4)/5=0.2.

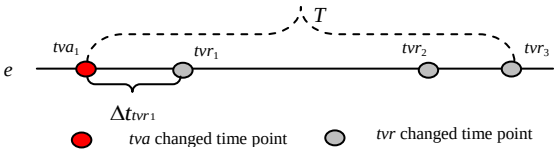


Fig.4. Example of learning *df*

C.Temporal Trend Similarity

Intuitionally we consider the following two aspects of temporal trend similarity from static and dynamic perspectives respectively: (1) Two references are more prone to the same real-world entity if they cover the same temporal intervals or are with the same timestamp in the same association or same attribute along the timeline. E.g., two person references' university study periods are both from 2004 to 2008. (2) Two references are prone to be the same real-world entity when the burst events or turning points happened synchronously, i.e, they have the similar temporal trends along the timeline. E.g., two author references both have published the largest number of papers in 2008. Another example, two researcher references both joined

Google at 2004 after obtaining their PhD degrees. In Fig. 5 we give a toy example of temporal trend of two author references a_1, a_2 along the timeline. The two curves indicate the number of papers published by the authors with time. And the waveform is composed of main bursty events of a1's education and professional experience such as PhD entrance,

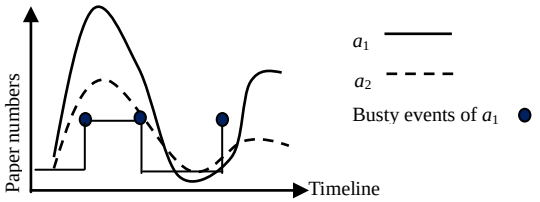


Fig.5. Temporal trend of two author references

PhD graduation and job hopping.

We calculate the temporal trend similarity of two references r_i and r_j on *tia*(denoted as $Sim_{trend}(r_i,r_j)_{tia}$), on *tir* (denoted as $Sim_{trend}(r_i,r_j)_{tir}$) as formula (9), (10) shown respectively. If the timespans ts_i, ts_j of r_i and r_j are equal or ts_i is inside ts_j, or the timestamps t_i, t_j of r_i and r_j are equal, we set the similarity score to be 1. If ts_i overlaps ts_j, we want the similarity score to represent the proportion of ts_i being inside of ts_j.

$$Sim_{trend}(r_i,r_j)_{tia} := \begin{cases} 1, t_i \ equals \ t_j \\ 0, otherwise \end{cases} . \quad (9)$$

$$Sim_{trend}(r_i,r_j)_{tir} := \begin{cases} 1, ts_i \ equals \ ts_j \\ \dfrac{|ts_i \cap ts_j|}{|ts_i \cup ts_j|}, ts_i \ overlaps \ ts_j \\ 0, otherwise \end{cases} . \quad (10)$$

So the normalized temporal trend similarity of two references r_i and r_j is computed as:

$$SimT_{trend}(r_i,r_j) = \frac{\sum_{k=1}^{|TIA|} Sim_{trend}(r_i,r_j)_{tia_k}}{|TIA|} + \frac{\sum_{k=1}^{|TIR|} Sim_{trend}(r_i,r_j)_{tir_k}}{|TIR|} . \quad (11)$$

V. TSE-Clustering Algorithm

In this section we first introduce the time-sensitive and evolution-based ER clustering algorithm TSE-Clustering in detail, and then present the semantic constraints and time-sensitive constraints used in TSE-Clustering.

A. Process of TSE-Clustering Algorithm

Time Labels of Cluster:. Each cluster c_i has two timestamps labels to denote the earliest and latest timestamps of references in cluster c_i. i.e., $c_{i.early}=Min\ (r_i.t)$, $c_{i.late}=Max\ (r_i.t)$ $r_i \in c_i$, and $c_{i.early}=c_{i.late}$ when $|c_i|=1$. And each cluster's timespan label (denoted as $c_{i.ts}$) is $c_i.ts=[c_{i.early}, c_{i.late}]$.

TSE-Clustering adopts hierarchical agglomerative clustering algorithm. We firstly adopt existed blocking technique to produce disjoint blocks to provide candidate references for the algorithm. Temporal information is also useful for blocking. E.g., if two movie references with same title but definitely different timestamps (year), they are probably two different movies produced by different directors or different nations. Then as the post-process of blocking, we use the temporal information of entities as a filtering factor and additional evidence to further reduce the size of larger blocks. The blocking algorithm T-Blocking is shown in Fig. 6. TSE-Clustering is only done on references in one block. At first, references of one block are sorted according to time order, and take each reference as a cluster $c_i=\{r_i\}$. Then compute cluster similarity scores, when similarity score of two clusters is more than threshold τ (set to 0.8 in our experiments, parameter setting evaluation see section 6.3), merge the two clusters and at the same time update timestamps labels of the new cluster accordingly. Then compute similarity scores of the new cluster with other clusters, iterative the above steps until no cluster similarity scores are more than threshold τ. At last we get the set of clusters each of which represents one real-world entity. The high level pseudo code for TSE-Clustering algorithm is shown in Fig.7.

```
Input:  R={r1,r2,…,rn}.
Output:  set of blocks B={ b1,b2,…,bi }.
//Stage 1: basic tradition blocking
1:  select blocking varibles;
2:  blocking R:  B={b1,b2,…, bn};
//Stage 2: post-process filtering
3:    for each block bi do
4:      if |bi|>size then
5:        select filtering factor fi;
6:      filtering bi according to temporal information;
7       end if
8:    end for
9:  return set of blocks B={b1,b2,…,bi};
```

Fig.6. T-Blocking algorithm

TSE-Clustering algorithm includes two parts: initialization part (line 1-2) and iterative part (line 3-14). It iteratively merges the most similar two clusters into a new one step by step until the similarity drops below threshold τ. At any step L, the current cluster set C_L reflects the current belief about the resolution results. At line 6 before merging two clusters,

we add time-sensitive constraints besides traditional semantic constraints, if merging any two clusters violates these constraints then the similarity score between the two clusters is set to zero if merging them violates any semantic constraints and time-sensitive constraints.

Input: References in one block R={$r_1,r_2,\ldots, r_{|R|}$}. Threshold of SIM: τ.

Output: Set of clusters C={$c_1,c_2,\ldots,c_{|C|}$}.

//initialization:

1: $C_0 \leftarrow \{c_1,c_2,\ldots,c_{|R|}\}$;

2: L=0; //L is the iterative step number

//Iterative:

3: $L \leftarrow L+1$;

4: for all two clusters $c_i \in C_{L-1}$, $c_j \in C_{L-1}$ do

5: compute SIM(c_i, c_j) ; //find a pair of clusters that is the most similar

6: if $SIM(c_i, c_j)>\tau$ then

7: if satisfy semantic constraints and time-sensitive constraints then

8: new cluster $c_{ij} \leftarrow Merge\ (c_i, c_j)$;

9: Update cluster's time labels $c_{i.early}$, $c_{i.late}$ and $c_{i.ts}$;

10: $C_L \leftarrow C_{L-1}-\{c_i,c_j\}\bigcup\{c_{ij}\}$;

11: else $SIM(c_i, c_j) \leftarrow 0$;

12: end if

13: end if

14: end for

15: return C_L

Fig.7. TSE-Clustering algorithm

B. Semantic Constraints and Time-sensitive Constraints

Semantic constraint is a domain dependent or application dependent rule that must be satisfied by two references during the clustering process to guarantee they are the same entity. For example, in bibliographic domain though two author references' name 'X. Dong' and 'X. L .Dong' are very similar, if they are co-authors of one paper, then they can not the same entity. Similarly in health care domain, two doctors with very similar names in one surgical operation cannot be the same person.

Time-sensitive constraint is the temporal rule enforced on two clusters to guarantee they can be merged during the cluster process. Currently we consider the following time-sensitive constraints in TSE-Clustering.

Time Continuity Constraints denoted as $Ccont(c_i,c_j)$. Continuity of references of the same real-world entity often can be observed. Take the complete DBLP data set as an example, two same name author references with big time gap (e.g., $|r_i.t-r_j.t|>80$ year) is impossible the same author. So we define two types of time continuity constraints: (1) $Ccont_1(c_i,c_j)$: $\forall c_i,c_j \wedge c_j.early \geq c_i.late \wedge c_i.ts \bigcap c_i.ts=\phi$, $Ccont_1(c_i,c_j)$ satisfies $c_j.early-c_i.late \leq n,\ n>0$. (2) $Ccont_2(c_i,c_j)$: $\forall c_i, c_j$, $Ccont_2(c_i,c_j)$ satisfies $max(c_i.late, c_j.late)$-$min(c_i.early, c_j.early) \leq m, m>0$ .Where n, m can be set by domain experts or system users (we set m=15, n= 80 in our experiments).

Time Currency constraints denoted as $Ccurr(c_i,c_j)$. Timestamps correlated with some attribute values of the same entity must satisfy a partial currency order. For example, for each author, the status value can only change from working to retired, from retired to deceased, but not from deceased to working or retired. So we define $Ccurr(c_i,c_j)$ as : $\exists r_i \in c_i$, $\exists r_j \in c_j$, $r_i.tva.t \leq r_i.tva.t$ $Ccurr(c_i,c_j)$ sat. $r_i.tva.value \prec_{tva} r_j.tva.vaule$, $tva \in TVA$.

VI. Experiments

First we give the data sets used in our experiments, parameter setting, and then give the experimental results and analysis. We conduct the experiments on a Windows machine with a 3.5 GHz Core i3-4150 Intel CPU and 8GB RAM.

A. Data Sets

We evaluate our approach using two data sets: 1) real data set DBLP. We extract authors and their papers and venues from the four research areas: DB (data base), DM (data mining), IR (information retrieval), and AI (artificial intelligence). And we manually extend some attributes values and associations with temporal information. 2) Synthetic data set (denoted as SynDS) from Cora, Google Scholar, and ACM Digital Library. We have inserted some artificial errors manually. Table II is the statistics of experimental data sets. Both data sets are divided into two parts respectively, one part is used to learn *aec*, *rec* and *df*, and the other part is used to test TSE-Clustering.

TABLE II. STATISTICS OF EXPERIMENTAL DATA SETS

Data Set	#references	#entities	Timespan(Year)
DBLP	19410	700	[1990,2012]
SynDS	18700	900	[1974,2008]

Evaluation Metrics: We compared pairwise matching decisions with the ground truth and measured the quality of

the result by *Precision* (*P*), *Recall* (*R*), and *F-measure*. The set of false positive pairs is denoted as F_p, the set of false negative pairs as F_n, and the set of true positive pairs as *T*. Then, $P = \frac{T}{T+F_p}$, $R = \frac{T}{T+F_n}$, and $F\text{-}measure = \frac{2PR}{R+P}$.

B. Parameter Settings

In experiments, β in formula (1) is set to 0.7. The intuition here is that attribute similarity and relational similarity is more important than the temporal trend similarity. We vary value of β to demonstrate the effect of β to the results in section 6.3. And we give a simple strategy to decide α.

$$\alpha = \frac{|attributes|}{|attributes| + |relations|} \quad (12)$$

Where |attributes| is the number of attributes used in similarity computation; |relations| is the number of relations used in similarity computation. E.g., if 5 attributes and 3 relations of an entity are used for similarity computation, then $\alpha = 5/(5+3) = 0.625$, $1-\alpha = 0.375$.

Similarly, μ, γ in formula (2) and formula (4) is set according to $|TIA|/|TIA|+|TVA|$, $|TIR|/|TIR|+|TVR|$ respectively.

And the thresholds $\theta_{high}, \theta_{low}$ are set as follows: $\theta_{high}=0.8$, $\theta_{low}=0.6$. We vary parameters θ_{high} and θ_{low}, and present some results in Section *C* to demonstrate robustness.

Temporal model learning: We divide the data set into two disjoint partitions of equal size. We learned temporal models from one partition at a time and used them to test with the remainder of the data set.

C. Different Parameters

We do experiments to verify the parameter settings on author entities. Firstly, we changed thresholds θ_{high} and θ_{low} for attribute similarity and got very similar results when $\theta_{high} \in [0.7, 0.9]$ and $\theta_{low} \in [0.5, 0.7]$. So we set θ_{high} as 0.8, θ_{low} as 0.6. Secondly, we applied different values of weight β ($\beta \in [0.1, 0.9]$) and the experimental result is shown in Fig. 8. And it shows the average *F-measure* score arrives at the maximum value when β is around 0.3. So we set β as 0.3.

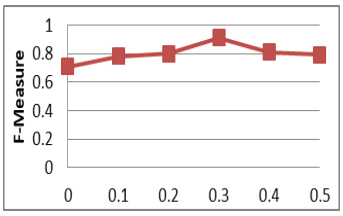


Fig. 8. Results of varying β values

Finally, we study the impact of the parameter τ to the performance of TSE-Clustering. Fig.9 shows the average F-measure when we vary τ ($\tau \in [0.5, 0.9]$). When τ is smaller than 0.8, the F-measure score increased steadily with the increase of τ. Then with τ is getting larger, the F-measure will decrease steadily. So we set value of τ as 0.8.

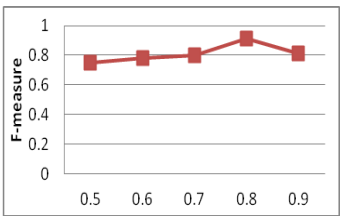


Fig. 9. Parameter setting about τ

D. Experimental Results

Effectiveness of TSE-Clustering: We adopt widely used evaluation metrics: pair-wise Precision, Recall and F-measure score. The experimental results on author, paper and venue entities are shown in Table III.

TABLE III. AVERAGE PRECISION, RECALL AND F-MEASURE ON DIFFERENT DATA SETS

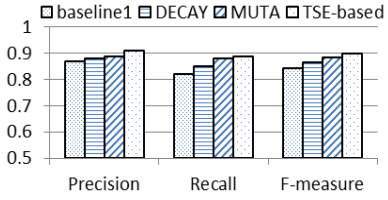
Data set	Entity class	Precision	Recall	F-measure
DBLP	Author	0.91	0.89	0.9
	Paper	0.93	0.88	0.904
	Venue	0.91	0.9	0.905
SynDS	Author	0.93	0.88	0.904
	Paper	0.91	0.89	0.9
	Venue	0.94	0.9	0.92

Comparisons of different similarity measures: We compare the similarity measures used in TSE-Clustering with other three similarity measures on author entities. The experimental results are shown in Fig. 10.

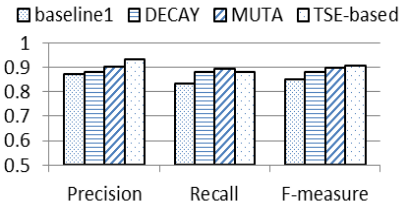
- Baseline1: traditional attribute similarity and relational similarity measure between two references, i.e., $Sim(r_i, r_j) = \alpha \times Sim_A(r_i, r_j) + (1-\alpha)\, Sim_{Rel}(r_i, r_j)$;
- DECAY: similarity measure used in [15] ;
- MUTA: similarity measure used in [16];

- TSE-based: Time-sensitive and evolution-based similarity measure used in TSE-Clustering.

As shown in Fig. 10, firstly, on both data sets all similarity measures considering temporal information do better than Baseline1. Secondly, similarity measure of TSE-Clustering outperforms those of DECAY and MUTA. And it shows that temporal trend similarity is benefit for improvement of ER results. Because it provides additional evidences for ER correctly. MUTA is better than DECAY by considering the probability of a given attribute value reappears over time.



(a)DBLP

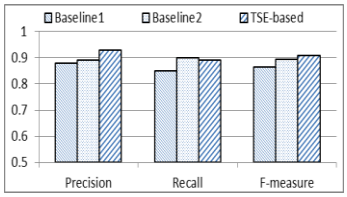


(b)SynDS

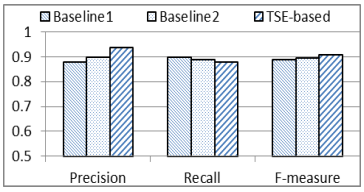
Fig. 10. Comparisons of different similarity measures.

Effect of Time-sensitive Constraints: We do experiments to evaluate the effect of time-sensitive constraints on author entities of two data sets, the experimental results are shown in Fig. 11. Currently we only consider the following two semantic constraints because sometimes they may produce false negative: 1) Co-authors of a paper must be different persons; 2) Two persons with the same first name but different last name are different persons. From Fig. 11 we can see that on both data sets Baseline2 outperforms Baseline1 by considering semantic constraints. And TSE-based obtains higher average *F-measure* scores than both Baselines. This suggests that the time-sensitive constraints have an important role to play on ER results.

- Baseline1: not considering semantic and time-sensitive constraints;
- Baseline2: only considering semantic constraints;
- TSE-based: semantic and time-sensitive Constraints used in TSE-Clustering.



(a) DBLP

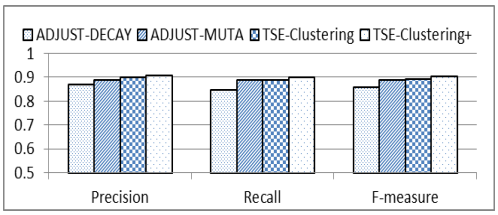


(b) SynDS

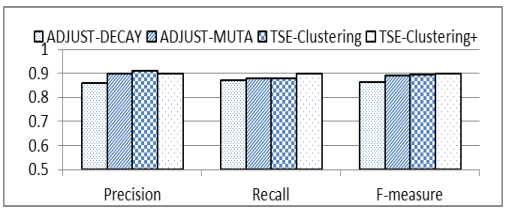
Fig. 11. Effect of time-sensitive constraints.

Comparisons of different clustering methods: We compare the following clustering methods with TSE-Clustering on author entities. The experimental results are shown in Fig.12. From the results we observe that on both data sets TSE-Clustering based methods achieve higher *F-measure* scores than ADJUST-DECY and ADJSUT-MUTA. And we can also see that TSE-Clustering+ achieve better results than TSE-Clustering due to adopting group average, but it will cause low efficiency compared to simple link strategy of TSE-Clustering.

- ADJUST-DECAY: temporal model and adjusted binding algorithm proposed in [15] ;
- ADJUST-MUTA: temporal model proposed in [16] and adjusted binding algorithm proposed in [15];
- TSE-Clustering+: based on TSE-Clustering, adopting group average when computing similarity of two clusters.



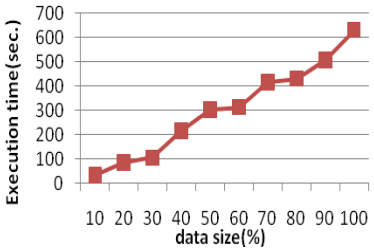
(a)DBLP



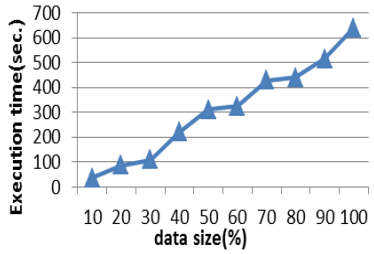
(b)SynDS

Fig. 12. Comparisons of different clustering methods.

Scalability of TSE-Clustering: To test scalability of TSE-Clustering by running it with different fractions of data sets, we divide the two data sets into 10 subsets with similar sizes without splitting entities respectively. The average execution times are reported in Fig.13. We observe that TSE-Clustering takes about 10 minutes to process fewer than 20k references on both data sets. The average execution times on both data sets grow nearly linearly with the increase of data sizes, showing scalability of TSE-Clustering.



(a)DBLP



(b)SynDS

Fig. 13. Scalability of TSE-Clustering.

VII. CONCLUSION

In this paper, aiming at ER in heterogeneous information spaces we have proposed a time-sensitive and evolution-based ER algorithm TSE-Clustering, which fully considers evolution effect and temporal information of entities. We capture the evolution effect of entities by introducing evolution coefficient, relation evolution coefficient when computing references similarity. Moreover evolution pair, evolution chain and evolution damping factor are introduced. We leverage temporal information to provide additional resolution evidence by introducing temporal trend similarity. Experimental results show our proposed ER algorithm achieves both high quality and scalability for ER in heterogeneous information spaces.

REFERENCES

[1] Ming Hua, Jian Pei. Aggregate queries on probabilistic record linkages[C]. In Proc. of EDBT 2012, 360-371.

[2] Ergin E., Min-Yen K., Lee D W, et al. Web based linkage[C].In Proc. of WIDM 2007, 121-128.

[3] Wang H Z, Fan W F. Object identification on complex data: a survey [J].Chinese Journal of computers, 2011, 34(10):1843-1852.

[4] X. Dong, A. Halevy, J. Madhavan. Reference reconciliation in complex information spaces[C]. In Proc. of SIGMOD2005, 85–96.

[5] Hanna Kopcke, Andreas Thor, Stefan Thomas et al. Tailoring entity resolution for matching product offers[C].In Proc. of EDBT 2012 ,545-550.

[6] Pei Li. linking records in dynamic world[C]. In Proc. of SIGMOD 2012 PhD Symposium, 51-56.

[7] Anitha Kannan, Inmar E. Givoni. Matching unstructured product offers to structured product specifications[C]. In Proc. of KDD 2011, 404-412.

[8] S. Minton, C.Nanjo,C.A.Knoblock. A heterogeneous field matching method for record linkage[C].In Proc. ICDM 2005,314-321.

[9] Z. Chen, D. V.Kalashnikv and S. Mehrotra. Exploiting relationships for object consolidation[C].In Proc. of SIGMOD 2005Workshop on Information Quality in Information Systems, 47-58.

[10] P. Domingos. Multi-relational record linkage[C] .In Proc. of KDD 2004 Workshop on Multi-Relational Data Mining,1-18.

[11] Indrajit B, Lise G. Relational Clustering for multi-type Entity Resolution[C].In Proc. of KDD 2005, 3-12.

[12] Indrajit B, Lise G. Collective entity resolution in relational data [J]. ACM Transactions on Knowledge Discovery from Data (TKDD), 2007, 1(1):5-es.

[13] S. Whang, H. Garcia-Molina. Joint Entity Resolution[C].In Proc. of ICDE 2012, 294-305.

[14] Nishita, Seikoh, Itoh, Motoki. Extracting relationship of meeting minutes generated by speech recognition system using entity resolution [J]. IAENG International Journal of Computer Science, v 43, n 3, p 284-289, 2016.

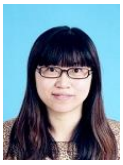
[15] Li P, Xin L D. Linking Temporal Records [J] .PVLDB, 2011, 4(11):956-967.

[16] Yueh-Hsuan Chiang, AnHai Doan, Jeffrey F.Naughton. Modeling Entity evolution for temporal record matching[C]. In Proc. of SIGMOD 2014, 1175-1186.

[17] Yueh-Hsuan Chiang, AnHai Doan, Jeffrey F.Naughton. Tracking entities in the dynamic world: a fast algorithm for matching temporal records [J]. PVLDB, 2014,7(6):469-480.

[18] Furong Li, Mong Li Lee, Wynne Hsu, et al. Linking temporal records for entity profile [C]. In Proc. of SIGMOD 2015.

.



Dan Yang received her M.S. and Ph.D. degrees in Computer Software and Theory from Northeastern University, China, in 2004 and 2013 respectively. She is currently associate professor of softeware school in University of Science and Technology Liaoning, China. Dr. Yang was a visting scholor in New Jersery Instittue of Technology, U.S.A from June 2015 to May 2016 supported by Chinese Scholarship Council of the Ministry of Education. Dr. Yang is a member of the CCF(China Computer Federation). Her research interests include semantic entity search and data integration.

Mo Chen received her M.S. and Ph.D. degree in Computer Software and Theory from Northeastern University, China, in 2008 and 2012 respectively. She is currently lecturer of school of computer science and engineering of Northeastern University. Dr. Chen is a member of the CCF(China Computer Federation). Her research interests include social networks and location-based service.