Shape detection of Structural Equation Models: A scalable approach using computer vision algorithms

Gerrit Merz
Institute of Information Systems and Marketing
Karlsruhe Institute of Technology
Karlsruhe, Germany
merz.gerrit@gmail.com

***Abstract*— This paper introduces the ever-expanding accumulation of knowledge in the scientific discipline Information Systems (IS) and shows one way to automatically identify the shapes of constructs within structuring equation models (SEMs). The results of this paper yield the conclusion that with a simple computer vision algorithm with further rule-based processes shapes can be identified and localized. Thereby, this paper marks the first step towards a scalable method for extracting information out of scientific papers.**

Keywords — structural equation models, shape detecion, computer vision, constructs identification, constructs location, cv2

I. INTRODUCTION

In the digital era, information is easily produced and accessible to everyone. Finding the right piece of information can therefore be a hardship and is a crucial skill if executed properly. However, even if the desired information can be found, it can sometimes be a tedious process to verify its meaning which highly depends on the context.

One such example can be found in the case of structural equation modeling (SEM). Many empirical studies are based on SEM as it allows scholars to quickly internalize concepts and theories (Hair, Sarstedt, Ringle, & Mena, 2012). In the scientific discipline, Information Systems (IS), SEM typically utilize – so called – constructs to observe and represent phenomena from the real world and thereby establish new or reinforce already encountered relationships and predictions (Mueller & Urbach, 2017). Scholars publish their supposed findings – in the form of constructs – to a wider audience for further refinement and as way of contributing to the academic advancement. Subsequently, researchers review published literature to discover existing findings and ultimately create new knowledge (Dann et al., 2019).

In IS, the prerequisite for theoretical advancement, to say the discovery of existing constructs and its overarching interdependences, is increasingly disturbed as missing guidelines encourage researchers to publish constructs and the results they contain in papers in an unstructured and non-standardized way (Dann et al., 2019). Another key problem scholars encounter while studying constructs is the two construct identity fallacies. In the case of the jingle fallacy, two different conceptual entities are assigned to the same construct name (Thorndike, 1904) whereas the jangle fallacy applies the reverse logic by assigning different construct names to the same perceived phenomenon (Kelley, 1927). As the number of scientific publications is rapidly increasing (Bornmann & Mutz, 2015), the above-mentioned problems need to be addressed.

In IS research tailored solutions on how to tackle the stated problems have either only be presented at theoretical level (see (Dwivedi et al., 2011) or (Larsen & Eargle, 2015) or lack the semantic incorporation of the analyzed constructs. Other researchers, however, explore the semantic context of constructs but the isolated paper by paper analysis makes it difficult to draw conclusions that are applicable beyond the paper itself (Dann et al., 2019).

To master the ever-expanding accumulation of knowledge in IS in a standardized and structured way, Dann, et al. (2019) propose to focus on the knowledge embedded within the constructs and the analytical and semantic relationships that exist by introducing the online platform DISKNET[1], which in a first step aims to allow the user to easily extract, discover and aggregate definitions of constructs as well as analytical and semantical relations.

According to DISKNET's webpage, the knowledge of 579 papers, 4705 constructs and 7653 relations has been aggregated. Currently, the aggregation as well as the

[1] https://disknet.iism.kit.edu/

construct detection and relation determination processes are executed manually. These processes can be described as follows: First, the about-to-be uploaded papers are manually scanned, then relevant structural equation models (SEMs) will be cropped and uploaded into the DISKNET system and lastly respective relations will be inserted into the provided interface. Given the increasing publication efforts and the abundance of data, this manual approach seems not only outdated, but also contains various negative implications. Not only is the manual extraction time-consuming, but the tediousness of the task itself also makes it very prone to error.[2] Furthermore, a manual approach hinders the scalability of the project and thereby undermines the basic idea of the DISKNET project: *To masterly handle the exponentially growing number of publications by capturing scientific knowledge in a machine-processable way.* Hence, the development of an algorithm that automatically identifies and localizes constructs and semantic relationships would be a self-evident next step in the context of the overarching goal of relevant paper detection and construct information extraction using ML models.

This paper tries to precisely address this issue by suggesting one possible solution by using a shape detection algorithm with further rule-based processes. The results of this paper are the following:

1. We provide an automatic solution to partly label data that can be used for further experiments.
2. We lay the groundwork for a further paper (Breitschopf, 2020) that simplifies manual post-labelling processes.
3. We show what advantages and shortages the presented algorithm has.

Overall, the main contribution of this paper is that with a simple shape detection algorithm with further rule-based processes desired constructs can be identified and localized with the example of SEMs.

The rest of the paper is structured as follows: Chapter 2 focuses on the idea and goal of this paper, presents the basic knowledge of SEM and gives an outlook what a potential code can and cannot achieve. Based on these explanations, chapter 3 introduces our own approach. The source code and the results obtained are presented and discussed in chapter 4 and 5. The conclusion in chapter 6 briefly reviews the progress of the work, the insights gained and the relevance for future work. The source code of the scripts as well as the relevant data sets are published on the university's internal Gitlab.[3]

II. The Task

A. Idea and Goal

In 1989, Ackoff (1989) introduced a concept in form of a pyramid which is called the Data-Information-Knowledge-Wisdom hierarchy. The nowadays taken-for-granted concept describes that data leads to information, information to knowledge which in turn leads to wisdom (Bernstein, 2009). Ackoff (1989) chose the form of the pyramid to express his idea as he posited that each below layer can be seen as a prerequisite for the above categories that need to be included. He elaborated further on his notion of the wisdom pyramid by explaining that "on average about forty percent of the human mind consists of data, thirty percent information, twenty percent knowledge, ten percent understanding, and virtually no wisdom" (Ackoff, 1989 p. 3).

The overall goal of DISKNET is to achieve scientific knowledge in a machine-processable way. Explained in the context of SEMs, DISKNET aims to implement an all-embracing same form of SEMs regardless of the unique characteristics of a discretionary structural equation model. Having established this, comprehensive SEMs across papers can be set-up and the exponentially growing number of publications becomes manageable. This, in turn, facilitates the process of reviewing existing literature for future researchers which is a precondition for extending the literature and ultimately for creating new knowledge.[4]

Having the concept of the Data-Information-Knowledge-Wisdom hierarchy in mind, this paper focuses on the recognizing part of the lowest layer of the pyramid (recognizing and getting the data) and tries to establish an automated approach that entails a shape detection algorithm with further rule-based processes for localizing and identifying constructs within SEMs.

The idea and the general goal in mind, the following subchapter first focuses on explaining the necessary basic concepts of SEM and then provides an overview with what kinds of SEMs an algorithm has to deal.

B. Structural Equation Modeling (SEM)

The usage of SEM can be described as a methodology for visualizing, estimating and testing a network of different relationships between variables (Rigdon, 1998). Within a structural equation model there exist nondirectional and directional connections among a set of measured (observed) and latent (unobserved) variables that account for the

[2] A possible error could occur in various ways for example by extracting the wrong information, misspelling or putting the right information in the wrong textbox.

[3] Link to Gitlab repository: https://git.scc.kit.edu/yn2099/research-model-annotation.

[4] According to (Dann et al., 2019), the process of screening existing literature is a necessary requirement for discovering existing constructs and what their relationships can tell us about the observations we want to unbundle.

hypothesized patterns (MacCallum & Austin, 2000). Generally speaking, SEM fulfills two goals:

1. to understand the patterns of correlation/covariance among a set of variables and
2. to explain as much of their variance as possible with the model specified (Kline, 1998).

The scientific field of SEM contains various subtleties that go far beyond the scope of the basic knowledge considered necessary to understand this paper. Therefore, an extensive explanatory guide can be found in Kline (1998).
Nevertheless, in order to establish a common understanding, figure 1 illustrates the key parts which are used to build a structural equation model.

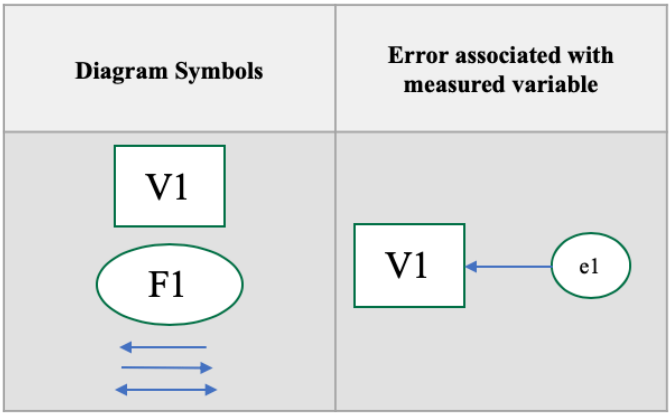

Figure 1: Key Parts of SEMs

On the left side, the following shapes – the basic diagram symbols SEMs consist of – are depicted: The rectangular called V1 represents the measured (observed) variable. The oval shape, F1, stands for the latent construct (unmeasured variable). The arrows express direct relationships and can also illustrate covariance or correlation if shown with two arrowheads on each end of the line (Kline, 1998). Another basic symbol combination that can be found in SEMs is depicted on the right-hand side. There, the construct in the rectangular shape that represents the measured variable V1 is associated with an error which is denoted by the smaller (compared to the oval shape of the unmeasured variable F1) oval shape e1.
Having established a basic understanding, one can imagine that SEM is a powerful tool to illustrate various interdependences and relationships and therefore can become very sophisticated. An inspection of the already uploaded SEMs incorporated into the DISKNET platform have yielded six basic cases which are depicted and enumerated in figure 2 from A) to F) respectively.[5]

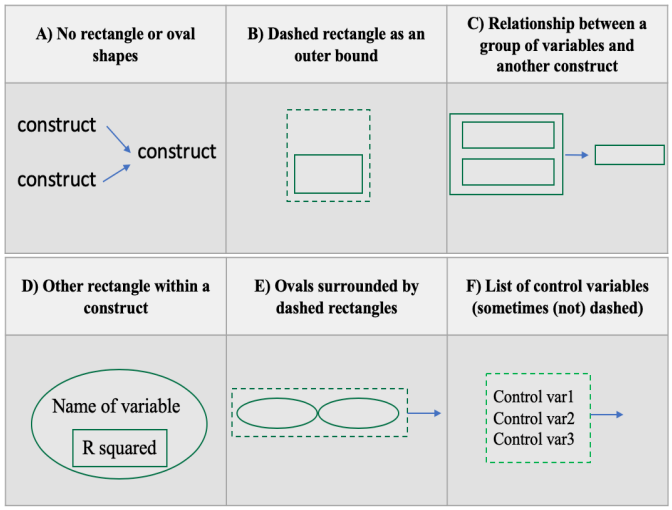

Figure 2: Basic case distinction of SEMs

Case A) shows a structural equation model that contains three constructs, but without any bounding boxes.
Another common case can be seen in B). Case B) depicts two rectangular shapes, however the bigger one is denoted by a dotted line and encircles the smaller rectangular.
Some SEMs show relationships between a group of variables and another construct. As shown in C), two smaller constructs have, according to the researcher, a relationship with another construct.
D) denotes a case in which the oval construct shape entails additional information, in this specific case the determination coefficient of the variable.
Similar to case C), E) suggests a relationship between a group of variables and (not shown in the picture) another group of variables or a single construct. What differentiates E) from C), however, is the fact that the ovals shown are surrounded by a dotted rectangle.
Last but not least, we encountered many SEMs in which the control variables were depicted in a non-uniform way. In the example of F) the 3 control variables surrounded by a dotted line are shown. However, in a further series of cases, the control variables were not encircled at all or instead with a solid line.

C. Code Detection Outlook

Looking at the different cases, we expect that computer vision algorithms will have certain problems with identifying the constructs. We expect problems to occur in the following 3 scenarios which are inferred from the cases depicted in figure 2:

[5] Figure 2 should be understood as a rough illustration of the different cases that we encountered. In reality, however, there were many more special cases and subcases. Because of their, partly, low frequencies and special subtleties, these cases have not been addressed in figure 2.

1. For instance, it is not possible for a simple shape detection algorithm to detect constructs in case there are no shapes around the construct (A). However, in order to train a ML model, the missing rectangular or oval shapes that enclose constructs are needed.
2. Also, when a shape which is not representing a construct lies within an actual construct, it could lead to false positives (see cases B), C), D) and E)).
3. The same applies to general shapes that indicate for instance the legend of the model (F).

Even though exceptions as the ones mentioned above could be caught to a certain extent by using filters or manual rule-based postprocessing of the constructs, the variety of cases a potential computer vision algorithm is confronted with makes the necessity of re-checking the annotated SEMs inevitable. This raises therefore the overall question which effort should be put into implementing all these rules in a manual rule-based code when the necessity of re-checking every image manually cannot be avoided.
Having seen the various different cases of SEMs and the potential problems that might come with it, chapter 3 explains the steps on how we tried to tackle the problem and shows the algorithm used to achieve it.

D. Side Note

As this paper focuses only on the technical concepts that are needed to implement a shape detection algorithm with further rule-based processes for construct identification and localization, a numerical overview, of how successful the algorithm is, is not provided. A thorough overview of the data set can be found in a subsequent publication (Breitschopf, 2020). The subsequent paper can be seen as a further study towards the overarching goal proposed by Dann et al. (2019) that is based on the acquired results in this paper.

III. Our Approach

Having seen the various different cases of SEMs and the potential problems that might come with the construct detection in the previous chapter, chapter 3 describes the idea of our shape detection algorithm with further rule-based processes to automatically identify and localize constructs using SEMs as the underlying data. The steps of our algorithm are illustrated in Figure 3 which shall serve as an orientation throughout this chapter.

In order to detect constructs in the SEMs' images provided, we employ the library OpenCV[6] (Open Source Computer Vision Library) which is commonly used for computer vision applications and as a machine learning software library. The pseudocode in figure 3 can be described as follows:

Using OpenCV to detect constructs (detect_constructs.py)

```
Data: Images of Structural Equation Models

For model in Data:
    Find contours using cv2
    For contour in contours:
        If contour's height and width are bigger than 30px and smaller than 70% of image:
            If contour is rectangle, circle or ellipse:
                contour is construct
```

Figure 3: Pseudocode of OpenCV script

First of all, each structural equation model is converted into a binary image in order to improve contour finding accuracy. In this context, a contour can be defined as a curve joining all the continuous points (along the boundary) which have the same color or intensity. Once contours are detected, it is verified whether the identified contours are potential constructs. For that, we first filter out contours according to the size, meaning it will be checked if the height or width of the contour has an adequate size (no construct if a contour is too small or too big). Having filtered out certain too small or too big contours out of the set of potential constructs, the algorithm applies the concept of a polygon curve[7], meaning that for the remaining contours a polygonal curve will be approximated. As each polygonal curve is specified by a certain number of points (e.g. A rectangle for instance is a polygon with 4 points), we can employ this information to tailor the detection of contours to our individual needs. We do this by specifying that we are only interested in contours that must have exactly 4 points or have more than 6 points and possess a convex shape as we want to detect oval or circle-like shapes as well. The exact-4-points condition helps us to filter out triangles and lines and the more-than-6-points-and-convex-shape filter enables us to detect oval shapes and circles. By approximating the polygonal curve, the code depicted in figure 3 leaves us with a handy and simple algorithm that can be used (as a first step) to detect potential constructs.

IV. Source Code

The source code is organized in one repository which entails the main script of this paper (detect_constructs.py), the main script of the subsequent paper (api_calls.py) that directly

[6] https://docs.opencv.org/.
[7] The concept of a polygon is used in geometry and describes a "plane figure that is described by a finite number of straight line segments connected to form a closed polygonal chain or polygonal circuit" (Wikipedia, 2020).

addresses problems and continues with the progress achieved here (Breitschopf, 2020) and some additional scripts that have not been included in this paper in accordance with our supervisor. The code is published within the university's internal Gitlab[8]. Within the repository, a more detailed and technical explanation of the source code and appendix is ensured via the attached README file. The additional scripts written in order to also detect relations between constructs can be found in Gitlab and the basic idea is depicted in the appendix. Further improvements of the algorithm were not conducted as discussions with our supervisor yielded the conclusion that focusing on an interface solution that alleviates the manual post-labelling process would outweigh the gained performance increase by catching all special cases (see figure 2) with further rule-based processes.

V. Results

With our approach we achieve overall solid results (see Breitschopf (2020) for a numerical overview), especially when taking the simplicity of the algorithm into account. Simple models with few constructs and clear relations have been detected reliably. We must consider that at this point the algorithm only detects the shape of the constructs, not yet the name of the construct itself. But especially detecting the bounding box of the constructs is crucial for instance when training a more flexible machine-learning (ML) model at a later stage. Compared to simple shape detection tools on the market, this paper's approach has the advantage that the code can be easily adjusted according to the needs of the researchers.
However, as expected, problems occurred in the following scenarios:

1. In case the image of the model has a low resolution, the algorithm can't detect any shapes.

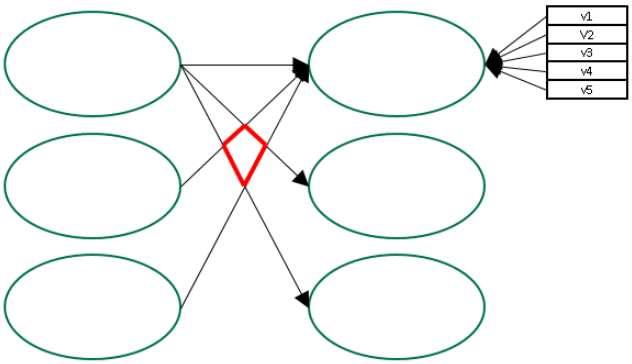

Figure 4: Detection error

2. The approach of approximating the detected shape with a polygonal curve and using the number of points of that curve as an indicator for a construct contains weaknesses, as depicted in figure 4 above. The highlighted red shape in the center of the figure represents a polygonal structure with 4 points. Also, the shape is large enough to not get filtered out by the height and width filters. Hence, the algorithm detects that shape as a construct.
3. The figure also shows another deficit of the algorithm. Looking at the constructs v1 to v5, the shapes around the constructs have almost the same height as the characters. Hence, the height filters will have the effect that those constructs will not be detected as constructs.
4. Other problems that occurred were that the algorithm detected some of the constructs twice.

Possible explanations why the algorithm could not detect some constructs will be given together with the numerical overview in the subsequent paper (Breitschopf, 2020).
To overcome the stated problems and further increase the performance of the detection algorithm, it would be possible to include other rule-based filters for post-processing detected constructs. For instance, a rule that filters out detected constructs within another construct could be implemented easily. However, since this paper is only one part of the overarching goal to extract knowledge from SEMs automatically using ML models, the effort to implement all these rules manually outweighs the gained performance increase because for training a ML model all SEMs would need to get re-checked manually and potentially re-annotated again.

VI. Conclusion and Future Work

This paper introduced the implications of the ever-expanding accumulation of knowledge in IS and showed one way to automatically identify the shapes of constructs within SEMs.
It can be seen as the starting point to come up with a ML model that automatically extracts knowledge from SEMs. The provided algorithm, data and insights facilitates further research and helps to reduce the manual effort in building a complex information extraction model as the already simple automatic detection of the shapes of the constructs reduces the annotation effort.
Furthermore, this paper also contributes to the existing literature by showing that a simple shape detection algorithm with further rule-based processing can be very suitable for identifying and localizing constructs and its relations from

[8] Link to Gitlab repository:
https://git.scc.kit.edu/yn2099/research-model-annotation.

SEMs. In that case, the algorithm would need to be extended so that it can handle more exceptional cases. Furthermore, the field of application is not only limited to SEMs. In all kinds of models or images where certain shapes occur, our approach could be suitable.

Since this paper is part of a bigger study there will be further research and papers related to that topic. Subsequent scholars should focus on how to facilitate manual post-labelling processes in a next step in order to come up with a final dataset that can be used for building a complex ML model. One such approach can be found in Breitschopf (2020).

VII. Bibliography

Mueller, B., & Urbach, N. (2017). Understanding the why, what, and how of theories in IS research. *Communications of the AIS*, pp. 349-388.

Dann, D., Teubner, T., Meske, C., Maedche, A., Mueller, B., & Funk, B. (2019). DISKNET – A Platform for the Systematic Accumulation of Knowledge in IS Research. *Fortieth International Conference on Information Systems*, 1-9.

Thorndike, E. (1904). An Introduction to the Theory of Mental and Social Measurements. *Science Press*.

Kelley, T. (1927). *Interpretation of Educational Measurements.* Oxford, UK: World Book Company.

Bornmann, L., & Mutz, R. (2015). Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references. *Journal of the American Society for Information*, 2215-2222.

Larsen, K. R., & Eargle, D. (2015). *Theories used in IS Research Wiki*. Retrieved from available at https://is.theorizeit.org/wiki/Main_Page; retrieved 20 February, 2020

Dwivedi, Y., Wade, M., & Schneberger, S. (2011). *Information Systems Theory: Explaining and Predicting Our Digital Society.* New York, USA: Springer.

Bernstein, J. (2009). The Data-Information-Knowledge-Wisdom Hierarchy and its Antithesis. *Proceedings North American Symposium on Knowledge Organization Vol. 2*, 68-75.

Ackoff, R. (1989). From data to wisdom. *Journal of Applied Systems Analysis 15*, 3-9.

Rigdon, E. (1998). *Structural equation modeling. In Modern methods for business research.* NJ: G. A Marcoulides (editor). Mahwah: Lawrence Erlbaum Associates, Publishers.

MacCallum, R., & Austin, J. (2000). Applications of structural equation modeling in psychological research. *Annual Review of Psychology, 51,*, 201-226.

Kline, R. (1998). *Principles and Practice of Structural Equation Modeling.* New York: The Guilford Press.

Hair, J., Sarstedt, M., Ringle, C., & Mena, J. (2012). An Assessment of the use of partial least squares structural equation modeling in marketing research. *Journal of the Academy of Marketing Science*, pp. 414-433.

Breitschopf, G. (2020). Minimzing the manual annotation effort of images: An example of annotating the constructs within Structuring Equation Models. *IISM*.

DISKNET. (2020, 02 28). Retrieved from https://disknet.iism.kit.edu/

KIT Gitlab. (2020, 2 28). Retrieved from https://git.scc.kit.edu/yn2099/research-model-annotation.

OpenCV. (2020, 2 28). Retrieved from https://docs.opencv.org/

Wikipedia. (2020, 2 26). Retrieved from https://en.wikipedia.org/wiki/Polygon

VIII. APPENDIX

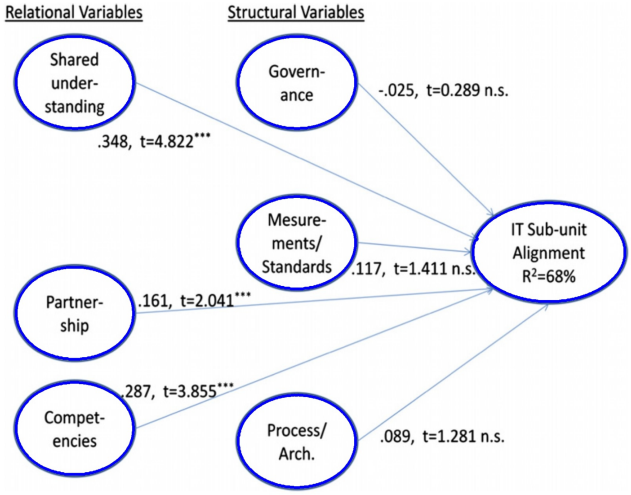

Figure 5: Correctly detected model

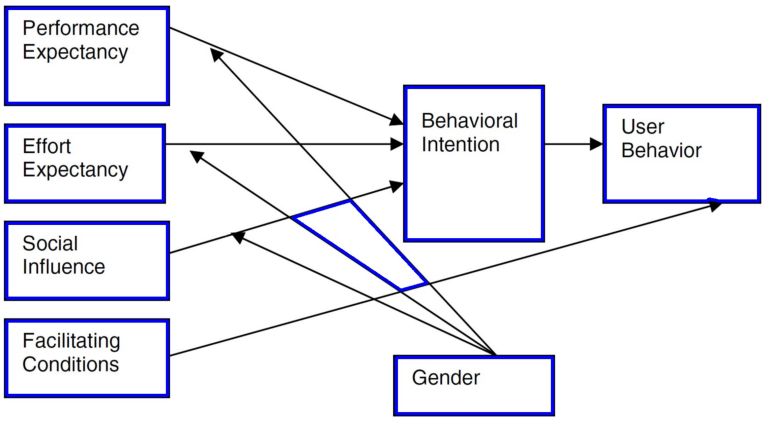

Figure 6: Model with one false positive

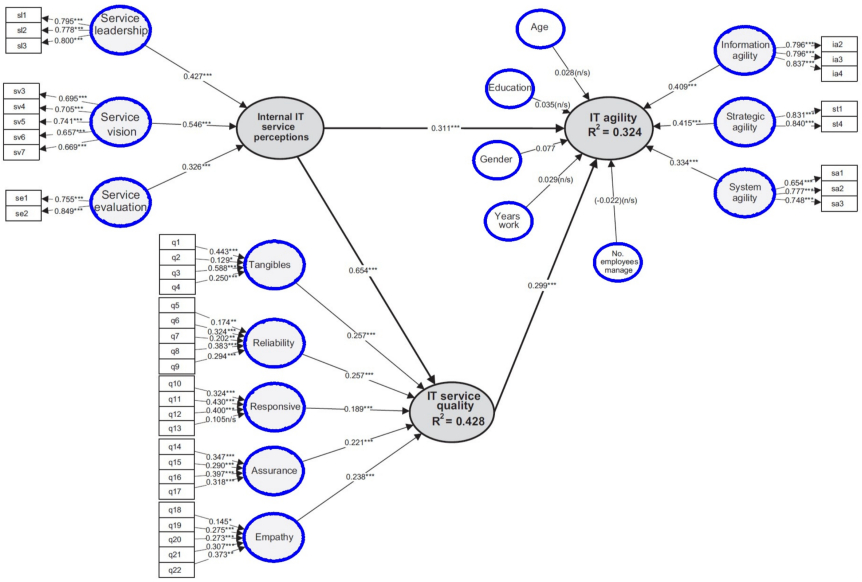

Figure 7: Model with many missing detections

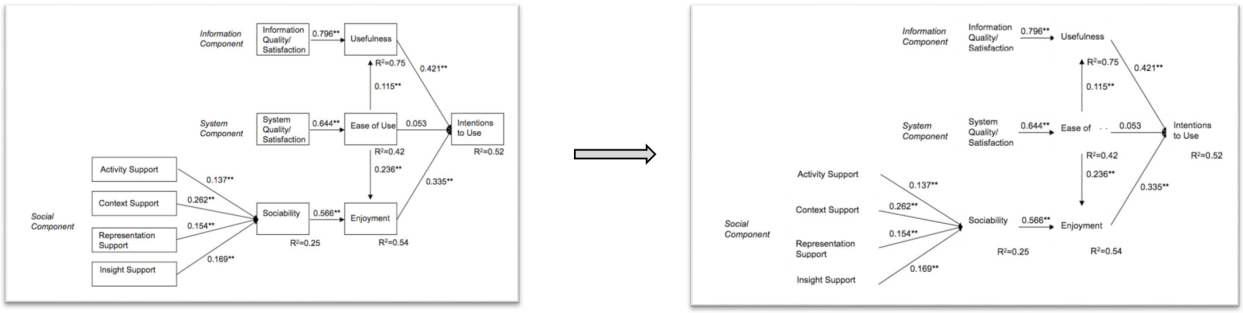

Figure 8: Relation Detection - Step 1

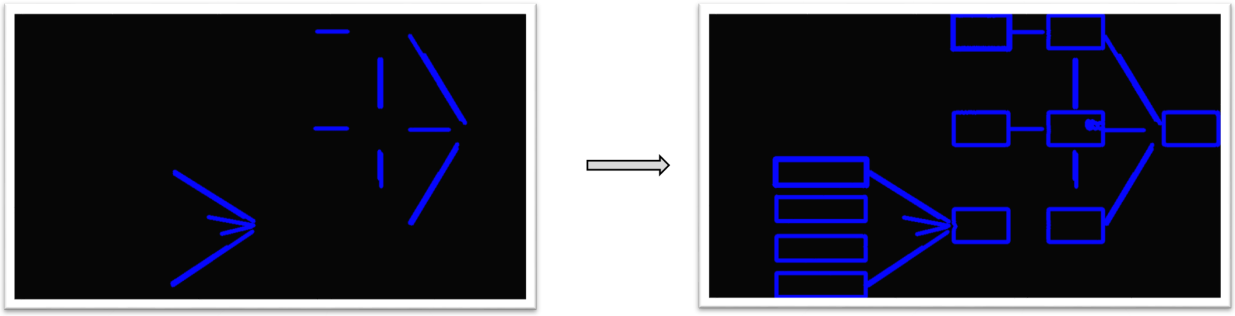

Figure 9: Relation Detection - Step 2

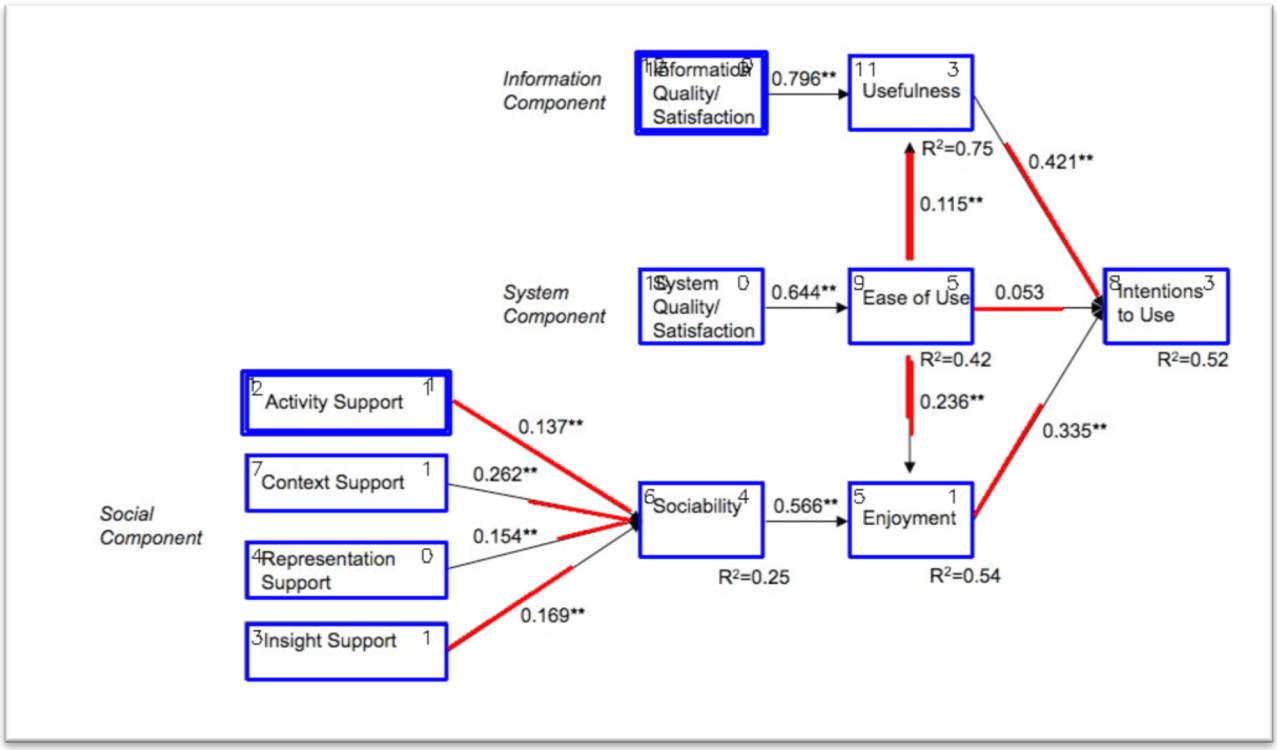

Figure 10: Relation Detection - Final Output