Automatic Design of Semantic Similarity Controllers based on Fuzzy Logics

Jorge Martinez-Gil

Software Competence Center Hagenberg GmbH
Softwarepark 21, 4232 Hagenberg, Austria
jorge.martinez-gil@scch.at

José Manuel Chaves-Gonzalez

University of Extremadura - Department of Computer Systems Engineering
Avda. de la Universidad s/n Caceres, Spain
jm@unex.es

Abstract

Recent advances in machine learning have been able to make improvements over the state-of-the-art regarding semantic similarity measurement techniques. In fact, we have all seen how classical techniques have given way to promising neural techniques. Nonetheless, these new techniques have a weak point: they are hardly interpretable. For this reason, we have oriented our research towards the design of strategies being able to be accurate enough but without sacrificing their interpretability. As a result, we have obtained a strategy for the automatic design of semantic similarity controllers based on fuzzy logics, which are automatically identified using genetic algorithms (GAs). After an exhaustive evaluation using a number of well-known benchmark datasets, we can conclude that our strategy fulfills both expectations: it is able of achieving reasonably good results, and at the same time, it can offer high degrees of interpretability.

Keywords: Knowledge Engineering, Fuzzy Logic Controllers, Semantic Similarity Measurement

1. Introduction

The computation of semantic similarity has been traditionally considered as an important method in many areas of computer research since methods of this kind are very important for successfully addressing a number of complex problems [41]. Automatically determining a similarity score for a pair of terms or textual expressions based on their real meaning is a problem that has attracted a multitude of researchers and practitioners belonging to a number of distant disciplines due to the fact that it has a number of implications in many application-oriented fields of different nature: computer science, linguistics, translation, literature, etc. However, finding a solution is far from being trivial since textual expressions usually lack objective features for fair comparison.

To overcome this problem, current trends on semantic similarity measurement follow an approach based on the aggregation of scores retrieved from individual semantic similarity measures (ssm) that make use of a wide range of heuristics and external resources. In this way, it is possible to reduce the risk of relying on a single ssm operating within production environments. Moreover, this approach has proven to achieve good results in the past [15]. The rationale behind this way of working is very intuitive; if there are some specific ssm not being able to perform reasonably well for the particular comparison of terms or textual expressions, their effects can be blurred by others ssm that achieve better performance. In this way, an overall improvement can be achieved, or put another way, the risk of making a severe mistake is greatly reduced. In general, aggregation methods try to combine different measures to come to the final decision.

In the particular context of semantic similarity measurement, recent advances of machine learning for natural language processing [54] have achieved a considerable increase in accuracy. In fact, training machine learning algorithms on large textual corpora has emerged as a powerful approach performing as well as traditional methods obtained after many years of research and fine-tuning [40]. One of the reasons why such approaches has achieved such good results is its inherent adaption capability. Artificial Neural Networks (ANNs) can be automatically configured in order to optimize the recognition of complex patterns. However, some authors suggest that new approaches based on ANNs are hardly interpretable [4, 5, 18]. This means that these models tend to behave like black boxes, i.e. it is possible to provide them with an input and get an output, but without the opportunity to understand what has happened in the intermediate process.

Therefore, even though there are already a number of solutions for aggregating semantic similarity, there are still some gaps that make the results not yet totally satisfactory. One of these major gaps is that state-of-the-art approaches have not been designed to handle the notion of interpretability. To overcome this limitation, we have worked towards the automatic design of semantic similarity controllers, which are systems that analyze semantic similarity values in terms of logical variables and produces a meaningful score by means of a complex, yet human-understandable, aggregation strategy. The key to achieving such good levels of interpretability is given by the fact that their behavior can be explained using rules that are easily understandable to humans. Moreover, this kind of logic controllers allows modeling nonlinear functions in the same way that ANN can, since both are considered universal approximators [14]. The reason for that is that we can approximate any nonlinear function if the search-space is divided into enough fuzzy sets. In any case, the success of such an approach is usually determined by the selection of those fuzzy sets (including their boundaries and membership functions), and the appropriate choice of the rules and defuzzification method. In this work, we aim to do so by using a genetic algorithm (GA) to help us optimize the process. Thus, we can summarize the major contributions of this work as follows:

- We introduce our research towards the automatic design and development of semantic similarity controllers being able to be accurate enough but without sacrificing its interpretability. In fact, our approach is able to achieve results in line with the best approaches, and at the same time, explain how these results have been achieved according to the IEC 61131-7 norm [34].
- We evaluate our strategy for the automatic creation of semantic similarity controllers using a wide pool of representative methods and datasets for semantic similarity including those intended for the general purpose, geospatial similarity, and biomedical similarity.

The remainder of this work is organized as follows: Section 2 introduces the state-of-the-art in relation to the design of controllers based on fuzzy logics. Section 3 presents the technical preliminaries necessary to understand our contribution. Section 4 describes our technical approach for the design and development of semantic similarity controllers. Section 5 reports an empirical evaluation of our proposal using a wide pool of datasets for semantic similarity. Finally, we highlight the conclusions and future research lines that could be derived from this work.

2. State-of-the-art

Fuzzy logics provides a framework for the representation of knowledge that allows modeling of the imprecision inherent in the human cognitive processes. In this way, fuzzy systems can be designed to solve a wide variety of problems dealing with complex situations involving vagueness and uncertainty. The problem here is that it is usually necessary to ask domain experts for designing the complete system, and therefore, the process becomes usually expensive in terms of money, time and effort needed. For this reason, it is unrealistic that the most appropriate setup can always be provided by those experts. That is one of the main reasons for the emergence of methods that try to automatically build Fuzzy Logic Controllers (FLCs) [29]. However, the generation of fuzzy terms and rules being easy to understand and to be reused in a variety of problems have traditionally been, and still is, one of the most challenging problems for the research community [20].

Concerning the specific challenge of semantic similarity measurement [50, 52], there are some works addressing the issue of how to use fuzzy logics in order to measure the degree of semantic similarity of text expressions [49, 27, 64]. The reason is that fuzzy systems are very suitable for dealing with the uncertainty and the ambiguity associated with the human language. In fact, the development of fuzzy logic emerged from the need to provide a framework to capture some of the uncertainties associated with cognitive activities such as the language used by people. This makes fuzzy logics a useful tool for modeling complex scenarios by means of fuzzy terms and rules [36]. Therefore, we think that it can help us in designing a conceptual framework to deal with the challenge of measuring semantic similarity and its inherent problems.

On the other hand, and since we are especially interested in the interpretability of our solution, the scientific community has proposed a number of different techniques to handle the so-called interpretability versus accuracy tradeoff. In particular, the use of FLCs is broadly extended due to their great capability of considering many different criteria [18]. In this context, given the non-linear nature of the output, traditional linear optimization tools have several limitations. However, GAs have proven to overcome these limitations by being a reliable method for optimizing the fuzzy terms and the associated fuzzy rules [6]. As a result, there has been a growing interest on the part of the community in the development of algorithms for automatically tuning FLCs that exploit these bio-inspired computing methods in order to benefit from the good search capabilities they offer. In this particular context, genetic fuzzy systems have been the subject of in-depth research [2, 1, 12, 28, 35] in last decades. The rationale behind these systems is that most of the components of a FLC can be obtained by using a GA [60]. In addition, their capability to incorporate existing knowledge [51] is also a very interesting characteristic in certain cases where is mandatory to impose some constraints [47].

It is also worth noting that there are already a number of approaches to automatically design FLCs to solve various problems in different application domains; the basic approach consists of just considering parameter identification, i.e. proper adjustment of the parameters of the membership functions, constrained by a previously known rule structure. The great drawback of such approach is that the structure needs to be known before the identification process. Another way to that is the so-called Michigan approach [13]. In that approach, the chromosomes are individual rules and a rule set is represented by the entire chromosome population. The collection of rules evolves over time through the evolutionary strategy that means that the different fuzzy rules need to cooperate under the action of the GA in order to achieve optimal results.

In this work, we deal with a variant of the so-called Pittsburgh approach [62] whereby the complete rule set is encoded within the same chromosome. However, in our approach not only the rules, but the complete FLC is encoded within a chromosome. This allows for the simultaneous evolutionary learning of all components together with the goal of generating the best possible design, even though this implies a broader search space. A solution scheme like this was already proposed in [31]. However, the novelty of our approach is that it is the first attempt to overcome the limitations of traditional semantic methods by automating the design of FLCs being able to asses the semantic similarity of textual expressions. In relation to existing works in the field of semantic similarity, our approach has the advantage of being able to conciliate computational capabilities of GAs with the human way of working of fuzzy systems, which leads to a similarity aggregation strategy being able to achieve reasonable levels of accuracy, and at the same time, being easily interpretable by a human, as we will be detailed in the next sections.

3. Technical Preliminaries

In the context of this work, a ssm is defined as a function intended to map the likeness of textual representations into a real value in the real interval [0, 1]. This function ssm: μ_1 x $\mu_2 \rightarrow$ R associates the degree of likeness for the textual expressions μ_1 and μ_2 to a real value v in the range $[0, 1]$, whereby a value near to 0 means not similarity at all, and near to 1 indicates the absolute similarity of the textual expressions μ_1 and μ_2 [11]. This is mainly due to the fact semantic similarity judgment is not always true or false, but obtains a certain degree of plausibility, depending on how well it reflects the human way of judging. According to Ballatore et al., most of existing terms could be semantically similar, at least, to some limited extent. Therefore, their degree of semantic similarity should be represented as a real value, instead of defined by using just a binary category [8].

There is a fundamental core of ssm covering different features from the natural language. In this context, the exploitation of background knowledge sources such as dictionaries, thesauri, large text corpora, and so on are among the most used methods by researchers and practitioners in this field. However, the real problem can be observed when each of these methods suggests different scores for the same particular case. A number of methods have been proposed in order to smartly aggregate the results of different approaches. In the literature, it is possible to observe how many of them have succeeded by overcoming the traditional limitations from simple ssm [46].

In this context, aggregation can be formally defined as a function $aggr : [0,1]^n \rightarrow [0,1]$ that is often carried out by an aggregation operator. These operators are mathematical transformations aiming to aggregate information from two or more different sources. For example, the arithmetic mean and the weighted mean are some of the most well-known aggregation operators. The major difference between them is that the weighted mean allows giving more importance to the different inputs according to some predefined relevance. In the literature, there is an important number of different aggregation operators that differ in the characteristics and the data they can work with [24]. But aggregation operators should always allow observing the following mathematical properties:

- *Identity: For a single input x,* $f(x) = x$
- *Boundary condition:* For f involving initial values N times so that, $f(0,0,...,0) = 0$ and $f(1,1,...,1) = 1$
- *Monotonicity:* For any pair $\langle a_1, a_2, ..., a_n \rangle$ and $\langle b_1, b_2, ..., b_n \rangle$ of n-tuples such that $a_i, b_i \in [0,1]$ for all $i \in N_n$, if $a_i \leq b_i$ for all $i \in N_n$, then $f(a_1, a_2, ..., a_n) \leq f(b_1, b_2, ..., b_n)$.
- *Continuity*: f is a continuous function.

However, the problem here is that all of these traditional methods (means, medians, modes, weighted means, etc.) often do not work well in the context of semantic similarity because they are based on very short-sighted strategies, i.e. strategies that just consider the numeric values of the inputs instead of analyzing the relationships among them [49]. Therefore, it seems a good idea to rely on fuzzy logics to try to model a problem that allows handling in a natural and intuitive way some of the aspects associated with the human language. In particular, fuzzy aggregation operators have been used recurrently in order to implement advanced reasoning mechanisms capable of overcoming a broad range of short-sighted strategies.

In this respect, some examples of fuzzy aggregation operators are fuzzy integrals and weighted sums [10]. These operators are intended to solve a wide variety of group decision-making problems

[32]. However, it is very common to find situations where these aggregation operators are not adequate to deal with a problem, since it is not always possible to know the model in advance. For cases like this, the research community has a great deal of expertise in developing solutions based on some kind of fuzzy reasoning implemented by means of rules. These rules allow for incorporating knowledge to be represented into the fuzzy system [3]. In addition, one of the great advantages of fuzzy systems that make them very suitable for our problem is that the working mode can be easily adapted to take a number of variables as input, and produce an unique output, which makes it a natural solution to the problem of aggregating semantic similarity [48].

Finally, it is important not to forget the role of the defuzzification process, i.e. the way in which a resulting set can be transformed into a real number. This is mainly due to the fact the output of the inference phase consists of fuzzy sets, which is largely undesired in many applications as ours. So in order to achieve the real number that represents the result from the aggregation process, we need a method that can generate a score from the resulting fuzzy sets. Although a detailed discussion of defuzzification methods is out of the scope of this work, good overviews of defuzzification strategies can be found at [26].

Example. When considering the semantic similarity of the words *journey* and *voyage*, human judgment has identified a similarity of 0.96 between them. If we want to replicate human judgment by means of an automatic system, we can rely on different measures exploiting their own strategies. We have tried five, and we got that ssm1: 0.17, ssm2: 0.80, ssm3: 0.69, ssm4: 0.47 and ssm5: 0.82. However, it seems that none of them is able to provide a satisfactory result. It would be possible to try to calculate a kind of mean, but the discordant results do not usually allow a good final result to be achieved. Therefore, the solution is to use semantic similarity controllers, where each value is encapsulated (to some extent) in one or more classes that roughly correspond to the way humans handle natural language: similar, not similar at all, very similar (a.k.a. fuzzification). Then it is possible to check which class or classes are the most represented (a.k.a. fuzzy reasoning), and finally, it is possible to map the resulting class(es) into a value similar to the one we were looking for (a.k.a.

defuzzification). As a result, if the semantic similarity controller is appropriately designed, then it can be able to properly compute new similar cases.

Therefore, we want to build a kind of FLC, i.e. semantic similarity controller, being able to decide whether a pair of textual expressions could be considered semantically equivalent or not. However, the overall problem here is that it is often difficult or unrealistic for human experts to define the fuzzy terms and fuzzy rules for this problem. In order to appropriately deploy that controller, it is necessary to study an important number of aspects such as fuzzy terms, membership functions, overlapping thresholds, defuzzification methods, etc. Therefore, we propose to formulate the automatic design from scratch of such controller as an optimization problem whereby the goal is to obtain a configuration being able to increase the chances of reaching our goal. This goal consists of replicating the behavior of experts when deciding on the semantic similarity of textual expressions to be compared. In this particular scenario, GAs have proven to be a useful method for finding configurations of this kind [31]. Therefore, we propose to use GAs to automatically design a semantic similarity controller based on fuzzy logics being able to achieve at the same time good performance and good interpretability levels.

4. Automatic Design of Semantic Similarity Controllers

The capability to design logic controllers is one of the most important applications of the fuzzy set theory in order to obtain accurate and human-comprehensible automatic rule-based expert systems. Logic controllers are usually divided into several components, among which the following stand out: a database of fuzzy terms such as $\mu_{\widetilde{S}}(x)$ that states the membership of x in $\widetilde{S} = \left\{ \int \frac{\mu_{\widetilde{S}}(x)}{x} \right\}$ what is usually defined in the real interval [0, 1], i.e. $\mu_{\widetilde{S}}(x) \in [0, 1]$, and a non-empty set of fuzzy rules. The rationale behind this organization is that the fuzzy terms associated with the database can be used to characterize fuzzy rules. These terms are mathematically defined using membership functions that are formulated on basis of expertise or engineering needs. The correct choice of these terms plays an essential role in the success of the FLC's performance. At the same time, the FLC's behavior is characterized by a set of linguistic rules (a.k.a. fuzzy rules) based on expert knowledge. A fuzzy rule is a structure like IF (some conditions are satisfied) THEN (some consequences are inferred). Since the conditions and the consequences of these fuzzy rules are associated with the aforementioned fuzzy terms, it makes sense to study a solution that considers both of them at the same time. Moreover, it is necessary to impose some convenient constraints as a way of expressing additional domain knowledge. In our case, this domain knowledge is oriented to help to improve the interpretability levels in relation to other problem-solving schemes.

Therefore, the automatic design of our semantic similarity controller is a complex task that comprises the automatic identification of (a) the input and output variables, b) the database of fuzzy terms, (c) the fuzzy rule base, and (d) the defuzzification method. In addition, we are going to make use of a standard from the International Electrotechnical Commission for fuzzy control programming, namely IEC 61131-7 [34], in order to rely on a well-known framework aiming to facilitate a common understanding of the design of our FLC. Listing 1 shows us an example of how the different parts of a FLC should look like.

Listing 1: IEC 61131-7 norm

```
FUNCTION_BLOCK
VAR_INPUT
    <variable name> REAL;
END_VAR

VAR_OUTPUT
    <variable name> REAL;
END_VAR

FUZZIFY <variable name>
    TERM <name> := <coordinates> ;
END_FUZZIFY

DEFUZZIFY
    METHOD: <method>;
END_DEFUZZIFY

RULEBLOCK
    <operator>:<algorithm>;
    ACCUM:<accumulation method>;
    RULE <rule number>: IF <condition> THEN <conclusion>;
END_RULEBLOCK
END_FUNCTION_BLOCK
```

Listing 2 shows a real example of a semantic similarity controller with four inputs that has been optimized to resolve the Miller & Charles benchmark dataset [55].

Listing 2: Example of automatically designed controller (1 of 2)

```
FUNCTION_BLOCK Semantic-Similarity
VAR_INPUT
        ssm1 : REAL;
        ssm2 : REAL;
        ssm3 : REAL;
        ssm4 : REAL;
END_VAR
VAR_OUTPUT
        score : REAL;
END_VAR
FUZZIFY ssm1
        TERM poor := (0, 1) (0.117, 1) (0.234, 0) ;
        TERM good := (0.234, 0) (0.351,1) (0.468,1) (0.585,0);
        TERM excellent := (0.468, 0) (0.585, 1) (1, 1);
END_FUZZIFY
FUZZIFY ssm2
        TERM poor := (0, 1) (0.117, 1) (0.234, 0) ;
        TERM good := (0.234, 0) (0.351,1) (0.468,1) (0.585,0);
        TERM excellent := (0.468, 0) (0.585, 1) (1, 1);
END_FUZZIFY
FUZZIFY ssm3
        TERM poor := (0, 1) (0.117, 1) (0.234, 0) ;
        TERM good := (0.234, 0) (0.351,1) (0.468,1) (0.585,0);
        TERM excellent := (0.468, 0) (0.585, 1) (1, 1);
END_FUZZIFY
FUZZIFY ssm4
        TERM poor := (0, 1) (0.117, 1) (0.234, 0) ;
        TERM good := (0.234, 0) (0.351,1) (0.468,1) (0.585,0);
        TERM excellent := (0.468, 0) (0.585, 1) (1, 1);
END_FUZZIFY
```

Listing 3: Example of automatically designed controller (2 of 2)

```
DEFUZZIFY score
        TERM poor := (0, 1) (0.117, 1) (0.234, 0) ;
        TERM good := (0.234, 0) (0.351,1) (0.468,1) (0.585,0);
        TERM excellent := (0.468, 0) (0.585, 1) (1, 1);
        METHOD : RM;
        DEFAULT := 0;
END_DEFUZZIFY

RULEBLOCK No1
        AND : MIN;
        ACT : MIN;
        ACCU : MAX;

        RULE 1 : IF ssm1 IS excellent AND ssm2 IS good THEN score IS good;
        RULE 2 : IF ssm1 IS excellent AND ssm3 IS good THEN score IS good;

        RULE 3 : IF ssm2 IS excellent AND ssm3 IS good THEN score IS excellent;
        RULE 4 : IF ssm2 IS excellent AND ssm4 IS good THEN score IS good;

        RULE 5 : IF ssm3 IS poor AND ssm4 IS good THEN score IS excellent;
        RULE 6 : IF ssm1 IS poor AND ssm2 IS good THEN score IS poor;

        RULE 7 : IF ssm3 IS good AND ssm1 IS good THEN score IS good;
        RULE 8 : IF ssm3 IS excellent AND ssm2 IS excellent THEN score IS good;

        RULE 9 : IF ssm2 IS excellent AND ssm4 IS poor THEN score IS good;
        RULE 10 : IF ssm4 IS good AND ssm3 IS good THEN score IS excellent;

END_RULEBLOCK

END_FUNCTION_BLOCK
```

Our task here is to automatically design from scratch a IEC 61131-7 [34] program leading to overcoming existing aggregation strategies for semantic similarity. To do so, we are encoding each of the aspects of such program (membership functions, fuzzy rules, defuzzification method, operator and accumulation method) as genes in a chromosome. An appropriate choice and organization of genes into chromosomes is very important in terms of the effectiveness and efficiency of the evolutionary strategy. In this way, and in order to avoid an explosion of the number of genes necessary to encode a chromosome, we have to be very careful to avoid an over-saturation of parameters that makes the design process unfeasible due to its high computational cost. Therefore, in the scope of this work, we will refer to the design by just using just the Basic Level language elements mentioned in the IEC 61131-7 norm.

As a result, and unlike neural solutions based on machine learning, where the model needs to be trained using a vast network of nodes interconnected to each other by means of edges with an associated weight, this proposal allows obtaining a human-readable program that facilitates to study and understand the mechanism of aggregation of the different ssm. In order to do so, our evolutionary strategy will be additionally constrained by a set of good practices leading to the obtaining of a very interpretable FLC. Then, that FLC can then be put into operation with the certainty that it will ensure better results than could be obtained with single methods and/or short-sighted aggregation strategies.

4.1. The input and output variables

On the one hand, the input variables are a type of linguistic variable that can take the values of fuzzy terms present in the database. Concerning this type of variables, the user will identify the most suitable ones at the beginning of the process. The idea is that the user provides a wide and heterogeneous group of different ssm using different text corpora, dictionaries or thesauri so that the decisions made by the FLC can be supported by a wide range of viewpoints. Since most of existing methods provide results in the real interval [0, 1] (or in its defect these results can be easily normalized in this interval), these will be the values that will be fuzzified according to the linguistic terms of the database, so that $\widetilde{I} = \mu_1 Q\left(x_1\right) + \mu_2 Q\left(x_2\right) + ... + \mu_n Q\left(x_n\right)$, whereby μ_i is the fuzzy term associated with the transformation of x_i into the fuzzy set $Q(x_i)$.

On the other hand, our FLCs work with the so-called Mamdani fuzzy inference [45] what means that the result of the inference inside the FLCs will be a fuzzy set such as $\widetilde{O} = \left\{\int \frac{\mu_{\widetilde{O}}(v)}{v}\right\}$. Therefore, we need a way to get the output variable as a numerical value in the real interval [0, 1] representing the result of the process of aggregating the different ssm. In order to do that, the definitive result will be delivered by means of a defuzzification method over the resulting fuzzy set. The most suitable defuzzification method will be identified by the evolutionary strategy.

4.2. The membership functions

Fuzzy terms are defined based on membership functions so that: $\widetilde{T} = \left\{\left(x, \mu_{\widetilde{T}}\left(x\right)\right) \mid x \in U\right\}$. These functions are usually not complex since it is assumed that complexity do not improve precision in this context. Some examples of fuzzy terms are the memberships to WordNet[1] classes and other Knowledge Bases such as Wikipedia[2]. In theory, it is possible to use many points to define membership functions, but in practice, a wide range of membership functions can be defined by just making use of four points: left lower and upper corners, and right lower and upper corners. This makes it possible to design a wide range of well-known shapes: square, trapezoid, shoulder, triangle, singleton, etc. The real values provided by the input variables can then be fuzzified as the linear interpolation between the two adjacent membership function points. Figure 1 shows us an example of some types of membership functions that can be obtained with just four points.

In our approach, the four points that define the membership functions are coded as genes in the chromosome representing each possible FLC. Just like the rest of the elements, the final membership functions will be those that the evolutionary strategy determines are the most appropriate to obtain

[1] https://wordnet.princeton.edu/

[2] https://www.wikipedia.org/

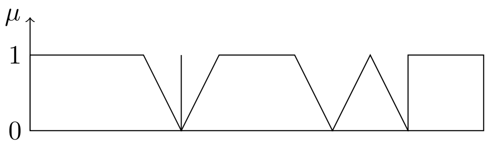

Figure 1: Examples of membership functions that can be defined with four points (shoulder, singleton, trapezoid, triangle, and rectangle)

the desired result according to the given input. In this way, if the designer decided to work with four different fuzzy terms (for example: very bad, bad, good and very good), we would need $4 \cdot 4 = 16$ different coordinates that would be automatically identified by the evolutionary strategy.

4.3. The fuzzy rule base

The fuzzy rule base stores the knowledge concerning the operation of the aggregation process. This knowledge is expressed by assigning relationships between fuzzy inputs and outputs. However, as mentioned above, it is unrealistic for an expert to be able to define these fuzzy rules in this context, and therefore we are preparing to generate them artificially. In order to avoid any assumptions on the rule structure, our approach is able to operate at symbolic level with some restrictions that limit the search space that would otherwise be unattainable. In our case, we allow the conditions of each fuzzy rule representing a statement or a combination of a maximum of two statements via the input variables, while the conclusion determines one single output. Other aspects that we will take into account are: we will only have one ruleblock, since having a many could hinder interpretability. Concerning the way to combine statements in the fuzzy rules, it is well-known that operators AND and OR must be used pair-wise in the following way: MAX corresponds to OR and MIN corresponds to AND [17]. In this work, we just allow AND combinations, since the OR combination can be achieved by means of two different rules.

It is also important to note that the controller being designed needs to calculate the degree of matching of the fuzzy rules, and then infer the resulting fuzzy sets. The inference system produces the same amount of output fuzzy sets as the number of rules collected in the rule base. These groups of fuzzy sets are aggregated by an accumulation operator, but they must be transformed into real values. Our accumulator operator will be also identified by the evolutionary strategy.

4.4. The defuzzification method

As a result of applying the previous step, we get always a fuzzy set. Therefore, we need a way to convert this fuzzy set into a real value. The defuzzification method will be also encoded as a gene into the chromosome. We are considering here those methods referenced in the IEC 61131-7 norm [34], i.e. Centre of Gravity, Centre of Area, Left Most Maximum, and Right Most Maximum as defined in [26]. The evolutionary strategy will automatically identify the most suitable one.

4.5. FLC identification

In our work, each of the parameters of the FLC is encoded as a gene into a chromosome. In general, the size of the population heavily depends on the nature of the scenario to be faced, but usually contains from dozens to thousands of possible solutions. The initial population is usually randomly generated, which it allows to look for the full range of possible solutions[3]. During each iteration, a portion of the existing population is selected to raise a new generation. Individual solutions are selected through an elitist process, where the best solutions are identified through the computation of the fitness (i.e. quality value for a solution). Our evolutionary strategy evaluates the suitability of each solution and selects the best solutions at each iteration. This is done through the maximization of the correlation coefficient between the human and artificial results.

In this context, the solution vector is highly dependent on the desired configuration for the semantic similarity controller which has to be automatically designed. This is mainly due to the need to encode the coordinates for the fuzzy terms, a number of rules, an accumulation operator, and the defuzzification process. Our solution is designed to accept up to N values, although the usual case is to work with configurations between 4 and 6 inputs. For example, in the specific case of a 4-input

[3] Although in principle there would be no restriction to use domain knowledge to start working with a good FLC

controller with 10 rules, the solution vector would consist of 68 values, which would be distributed as follows: 16 real values between 0 and 1 (4 coordinates for each of the 4 fuzzy terms), 50 discrete values for the fuzzy rule base (a maximum of 2 antecedents and 2 consequents and 1 conclusion for each of the 10 rules), 1 discrete value to identify the accumulation operator, and 1 discrete value to identify the defuzzification process.

Although there are many possible variants of the basic GA, we work here with a classic elitist strategy since it is what is best suited to our goal of maximizing both precision and interpretability. In fact, our approach consists of three major operations: evaluation of individual fitness, a compilation of the intermediate population through a selection procedure, and combination through crossover and mutation strategies. Anyway, we are performing a preliminary study to assess the specific configuration for each operation [39].

Algorithm 1 Pseudo-code for the evolutionary strategy to obtain the final FLC

```
procedure AUTOMATIC DESIGN OF FLC
    generationRandomFLCs (population)
    calculateFitness (population)
    while (stop condition not reached) do
        for (each individual of the population)
            parents ← selectionOfIndividuals ()
            offspring ← binCrossOver (parents)
            offspring ← randomMutation (offspring)
            calculateFitness (offspring)
            population ← updatePopulation (offspring)
        endfor
    endwhile
    return automaticallyDesignedFLC (population)
```

Algorithm 1 explains in pseudo-code how the whole process is performed; at the beginning, the chromosome population (i.e. the FLC population) is randomly initialized. According these initial parent chromosomes produce off-spring by the application of genetic operators: selection (i.e. identification of the genes to be chosen for evolution), crossover (i.e. combination of the information of two solutions from the population), and mutation (i.e. random change in a specific part of the solution). Selection is performed for the whole population. The operator selects which chromosomes should remain in the population and set up the crossover. Crossover and mutation are conducted separately on each part of the chromosome so there is no exchange of information between the parameters and the structure. For a more detailed explanation about the GA, please consult the original reference [30].

During the process, the evolutionary strategy is guided towards the goal of maximizing the results of the FLC that is being considered in relation to the results given by a human for the same input data set. This is done by comparing the degree of correlation between the background truth (i.e. the set of past cases solved by the experts in the specific field and which are assumed to be accurate) and the results issued by the controller. As we wish some predictive capability, we must also consider cross-validation when calculating the fitness. The Pearson Correlation Coefficient between these two numerical vectors can be formally defined using the following formula (where x is the numerical vector representing the results from those cases solved by a human expert, and y the numerical vector representing the results of those cases solved by the computer):

$$r_{xy} = \frac{n \sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n \sum x_i^2 - (\sum x_i)^2} \sqrt{n \sum y_i^2 - (\sum y_i)^2}}$$

Concerning the implementation, the fuzzy engine chosen in order to test the fitness of each generated model has been jFuzzyLogic [17] as it combines a friendly programming interface with a rigorous implementation of the IEC 61131-7 norm.

4.6. Interpretability as a requirement

According to Cordon et al., accuracy has been during many years the major goal of researchers and practitioners in this area [20]. The problem is that this fact has made the resulting fuzzy systems a kind of black-box model which is hard to interpret. However, in recent years, the scientific community has come back to its origins when considering design techniques by formulating the problem as an optimization of the accuracy and interpretability at the same time [19].

In this context, it is clear that interpretability is a quality that is not easy to define or quantify. However, there is a number of works in the literature that remark the advantages of fuzzy logics in order to overcome the limitations of traditional logic to work with imprecise rules that are very descriptive and easily understood by people at the same time [3, 4, 23, 29]. Moreover, several works have addressed the problem of considering interpretability in the design of fuzzy systems. In the context of this work, we are going to assume that the interpretability requires at least[4]:

- *Small number of fuzzy rules.* Some empirical studies state that this amount should be preferably no more than ten [4]. The reason is that fuzzy systems with a large amount of rules are usually more difficult to interpret than those systems that only need a few rules.

- *Small number of fuzzy terms.* This aspect is key in the whole design process as both the accuracy and interpretability of our solution might rely on them. Having a sufficient number of fuzzy terms is of vital importance to capture nonlinear functions. However, good practices on interpretability suggest keeping this number small. In this work, we have decided to adopt a compromise solution, allowing to have the same number of fuzzy terms as input variables.

- *No overlapping of more than two membership functions.* This fact helps to enhance the interpretability of the model since it keeps the fuzzy terms distinguishable. Moreover, the relative position of the fuzzy terms has to be maintained.

[4] Including additional interpretability considerations would be trivial

- *Completeness of the fuzzy partitions.* The fuzzy partitions have to be complete. This completeness condition makes it possible to assign a clear and unique meaning to each fuzzy term.
- *Consistency of the fuzzy rules.* This means that the fuzzy rules of the FLC do not have to be inconsistent. An inconsistency problem happens when two different rules have the same conditions but different consequences.

So the GA takes into consideration these constraints in the following way: getting the number of rules and the fuzzy terms to be low is a trivial task, since we only have to indicate the maximum number of rule slots and fuzzy terms that we need when coding the chromosome. No overlapping of more than two membership functions can be achieved by forcing the lower coordinates of those functions to be chained to each other, and imposing that only two successive links of the chain can be overlapped. Completeness is guaranteed by forcing both position 0 and position 1 of the X-axis to be covered (Furthermore there can be no gaps as the lower coordinates of the membership functions are chained). Finally, when two fuzzy rules have the same conditions but different consequences, we assign to that solution the minimum fitness so that it does not have options to be reproduced.

5. Results

We summarize here the results from the experiments that we have performed. It is important to remark that the comparison between the scores from our automatically-designed semantic similarity controllers and the human judgments can be expressed as a correlation between two numerical vectors of the same size, whereby each position of the vector indicates an entry in the semantic similarity datasets. This means that we aim to obtain the degree of similarity between the results from our approach and human judgments. To do that, we have considered some of the most widely used datasets from a number of different fields: general purpose, geospatial similarity, and biomedical similarity.

The rationale behind this way to measure similarity is to compare the degree of correlation between an artificial and a natural solution using the Pearson correlation coefficient. Each of both solutions contains all the similarity scores associated with each particular case from the benchmark dataset.

The final result will be between the values -1 (human ratings and results from the proposed solution present an opposite correlation) and 1 (human ratings and results from the proposed solution present a perfect correlation). Obviously, our challenge here is to get a result as close as possible to 1, what means that our approach could properly replicate the way of thinking of the experts who initially solved the benchmark dataset. In addition, it is also necessary to perform a significance test in order to determine if the null hypothesis can be accepted or rejected in all the experiments performed. We estimate a threshold for the p-value parameter as the usual $5.0 \cdot 10^{-2}$, what means that achieving a statistically significant correlation can be interpreted so that there is less than a 5% chance that the given correlation happened by chance.

5.1. Experimental setup

It is important to mention that when solving each case from a benchmark dataset for semantic similarity assessment using artificial methods, we are always facing one of these three situations: a) false positive, i.e. a result which wrongly indicates that the two textual expressions are semantically similar, b) false negative which wrongly indicates that the pair is not semantically similar, and c) hit, i.e. result which correctly indicates whether the two expressions being compared are semantically similar or not. In this work, we look for the largest number of hits (or the lowest number of false positives and false negatives together), and to achieve this we have performed a preliminary study from which we have concluded that the configuration of our GA should be the following:

- Representation of genes (binary, real): **real**
- Population size [10, 100]: **50**
- Crossover probability [0.3, 0.95]: **0.5**

- Mutation probability [0.01, 0.3]: **0.09**
- Iterate over **100,000** generations

After having performed a parametric study consisting on experiments from 1,000 to 100,000 iterations, we have concluded that the best results are obtained on average after around 45,000 iterations. However, in order to ensure the proper functioning of the system for the most complex benchmark, and taking in consideration that no temporal restriction is required, we have established the value of the iterations at 100,000. In this way, this configuration ensures a good performance for the simplest benchmark, but also for the most complex one.

In addition, it is necessary to mention that the results reported in next subsections are the result of a cross-validation process with 80% of the instances for training and 20% for validation what is the most common split used in problems of learning non-linear metrics [38]. Moreover, the experiments have been performed in a computer with Windows 10 64-bit over a processor Intel Core i7-4790 at 3.60Ghz and 8 GB of RAM memory. As an example, we can mention that the automatic design of a FLC aiming to aggregate 4 ssm and needing 100,000 iterations takes about 12 hours of processor time. Please note that it is not just matter of the dimension of the solution vector and the size of the associated solution space, but the execution and the cross-validation also contribute with large overheads to the automatic design process.

5.2. General purpose semantic similarity

The first dataset that we use is the so-called Miller & Charles [55] which is the traditional dataset used by the community to evaluate research approaches focused on general scenarios. Table 1 shows us this dataset which is intended to measure the semantic similarity between terms belonging to a general purpose scenario, that is to say, terms that one can find in numerous everyday situations. Samples from this dataset range from pairs that do not look alike at all (rooster and voyage) to other pairs that are absolute synonyms (automobile and car). The values are included in the real interval

[0, 4] but can be very easily normalized in the real interval [0, 1] because the Pearson correlation coefficient is invariant against the linear transformation.

Wordpair	**Human**	**Wordpair**	**Human**
rooster-voyage	0.08	crane-implement	1.68
noon-string	0.08	brother-monk	2.82
glass-magician	0.11	implement-tool	2.95
chord-smile	0.13	bird-crane	2.97
coast-forest	0.42	bird-cock	3.05
lad-wizard	0.42	food-fruit	3.08
monk-slave	0.55	furnace-stove	3.11
shore-woodland	0.63	midday-noon	3.42
forest-graveyard	0.84	magician-wizard	3.50
coast-hill	0.87	asylum-madhouse	3.61
food-rooster	0.89	coast-shore	3.70
cementery-woodland	0.00	boy-lad	3.76
monk-oracle	1.10	journey-voyage	3.84
car-journey	1.16	gem-jewel	3.84
brother-lad	1.66	automobile-car	3.92

Table 1: Samples from the Miller-Charles benchmark dataset

For this dataset, we have achieved a correlation coefficient of 0.855. Figure 2 shows the distribution of the results for the different cases that make up the benchmark. The ssm chosen for performing the aggregation have been Jiang & Conrath [37], Leacock & Chodorow [42], Lin [44], and Resnik [58]. All these semantic similarity measures are based on the exploitation of human-compiled dictionaries what makes it possible to achieve a high degree of interpretability. As we can see, the behavior of our automatically designed controller is close to human judgment. There are many hits, and there are not serious false positives. However, the results are not perfect because there are some false negatives. For further information about the intermediate results, please refer to Appendix A.

5.3. Geospatial semantic similarity

The second dataset comes from the geospatial field. The so-called GeReSiD [8] covers a pool of geographic terms including almost 100 unique textual expressions that have been clustered into 50 unique pairs. The human judgments of semantic similarity have been collected separately on the 50

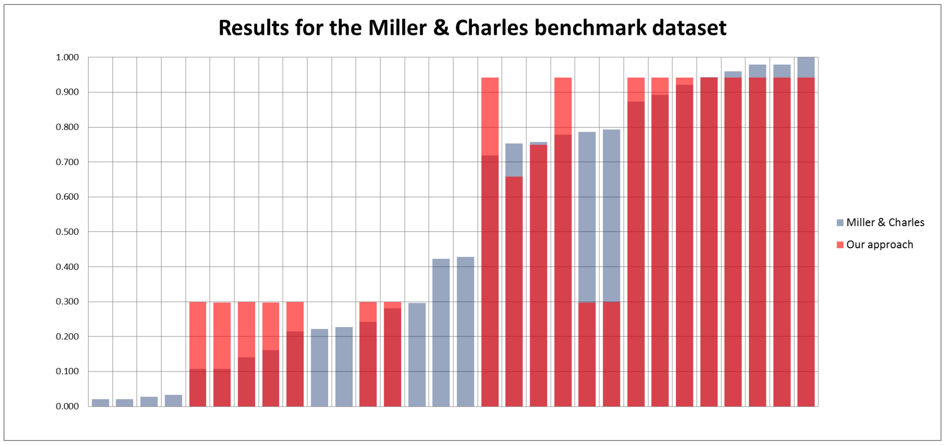

Figure 2: Results achieved over the Miller & Charles dataset

pairs. Table 2 shows us these 50 pairs which range from samples that are not similar at all (nursing home & continent) to other ones that are almost identical (motel & hotel) according to the human opinion.

For the GeReSiD dataset, we have achieved a correlation coefficient of 0.726. Figure 3 shows the result distribution that we have obtained. The ssm that we have chosen for being aggregated using our FLC have been UMBC and UMBC-STS [25], and LSA and LSA2 [21]. As it is possible to see the FLC does a great job of getting right the cases in which there is no similarity at all as well as the cases in which the similarity is almost total. However, the results are not absolutely perfect due to some cases of intermediate similarity. For having an overall view of the intermediate results, please refer to Appendix B.

5.4. Biomedical semantic similarity

MeSH dataset [56] is one of the most popular datasets for assessing semantic similarity of biomedical nomenclature. It is assumed that a method that is capable of achieving good results using this dataset should be able to support indexing and retrieval of biomedical articles stored in heterogeneous databases. This dataset is composed by a set of 36 word-pairs extracted from the MeSH ontology, and it ranges from pairs that are absolutely different (Anemia & Apendicitis) to other ones that are

expr1	**expr2**	**human**	**expr1**	**expr2**	**human**
nursing home	continent	0.0169	speed bump	car park	0.3893
political boundary	women's clothes shop	0.0208	sea	island	0.3914
greengrocer	aqueduct	0.0310	managed forest	significant tree	0.3992
interior decoration shop	tomb	0.0504	swimming pool	water reservoir	0.4174
water ski facility	office furniture shop	0.0517	industrial land use	landfill	0.4385
community center	stream	0.0579	mountain hut	hilltop	0.4897
city suburb	antiques furniture shop	0.0717	barracks	shooting range	0.5145
vending machine	gate	0.0806	church	historic ruins	0.5348
fashion shop	swimming spot	0.0847	glacier	body of water	0.5574
beauty parlor	fire station	0.0943	canal	dock	0.5943
football pitch	corporate office	0.1086	police station	prison	0.6107
panoramic viewpoint	race track	0.1240	tower	lighthouse	0.6168
bed and breakfast	school building	0.1393	administrative office	town hall	0.6209
shelter	agricultural field	0.1488	historic castle	city walls	0.6446
ambulance station	city	0.1542	restaurant	beverages shop	0.6496
arts center	bureau de change	0.1612	historic battlefield	monument	0.6680
supermarket	surveillance camera	0.2042	art shop	art gallery	0.7480
post box	town	0.2097	bay	body of water	0.7623
school	toy shop	0.2172	stadium	athletics track	0.7643
canoe spot	hunting shop	0.2354	tram way	subway	0.7643
office building	academic bookstore	0.2686	floodplain	wetland	0.7686
car store	cycling facility	0.2727	basketball court	volleyball facility	0.7807
heritage item	valley	0.2896	public transport station	railway platform	0.8115
city	railway station	0.3279	theater	cinema	0.8730
picnic site	stream	0.3689	motel	hotel	0.9037

Table 2: Samples from the GeReSiD benchmark dataset

absolutely equivalent (Chicken Pox & Varicella). Table 4 shows us this benchmark dataset.

For this dataset, we have achieved a correlation coefficient of 0.781. Figure 4 shows how the results for each sample are distributed. The ssm that we have chosen for being aggregated using our FLC are Path-based [57] Leacock [42], Adapated Lesk [9], and Resnik [58]. On this occasion, the FLC behaves very well in detecting cases of absolute and medium similarity, although it incurs in some false positives in samples that are not similar at all. Intermediate results are presented in the Appendix C.

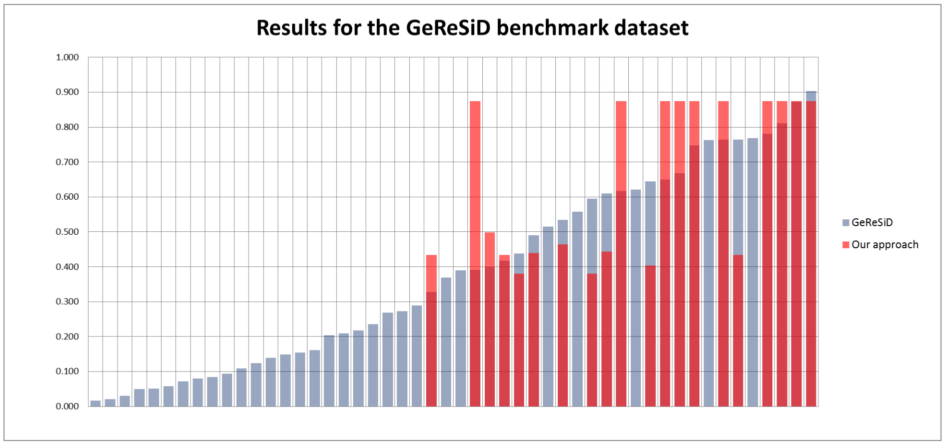

Figure 3: Results achieved over the GeReSiD dataset

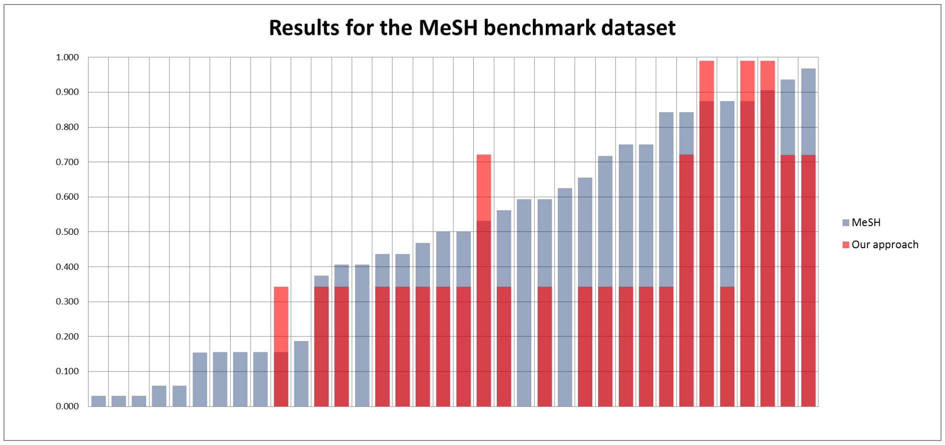

Figure 4: Results achieved over the MeSH dataset

ExpressionA	ExpressionB	Human
Chicken Pox	Varicella	0.968
Antibiotics	Antibacterial Agents	0.937
Measles	Rubeola	0.906
Pain	Ache	0.875
Malnutrition	Nutritional Deficiency	0.875
Down Syndrome	Trisomy 21	0.875
Breast Feeding	Lactation	0.843
Seizures	Convulsions	0.843
Carcinoma	Neoplasm	0.750
Myocardial Ischemia	Myocardial Infarction	0.750
Migraine	Headache	0.718
Ur.Tract Infection	Pyelonephritis	0.656
Failure to Thrive	Malnutrition	0.625
Vaccines	Immunity	0.593
Psychology	Cognitive Science	0.593
Hepatitis B	Hepatitis C	0.562
Pulmonary Stenosis	Aortic Stenosis	0.531
Hypertension	Failure	0.500
Lactose Intolerance	Irr. Bowel Syndrome	0.468
Adenovirus	Rotavirus	0.437
Hypothyroidism	Hyperthyroidism	0.406
Sarcoidosis	Tuberculosis	0.406
Otitis Media	Infantile Colic	0.156
Hyperlipidemia	Hyperkalemia	0.156
Bacterial Pneumonia	Malaria	0.156
Osteoporosis	Patent Ductus Arteriosus	0.156
Sequence	Antibacterial Agents	0.155
Acq. Immunno. Syndrome	Congenital Heart Defects	0.060
Dementia	Atopic Dermatitis	0.060
Meningitis	Tricuspid Atresia	0.031
Sinusitis	Mental Retardation	0.031
Anemia	Appendicitis	0.031

Table 3: Samples from the MeSH data set

5.5. Comparison With Existing Works

In principle, it seems reasonable to think that having good interpretability and high accuracy are contradictory aims, and one could think that reaching high levels of interpretability at the expense of poor accuracy does not seem to have a practical impact. Once that it is clear that our controllers use a model that has been specifically designed to produce results that can be interpretable, it is time to compare the accuracy with the classical solutions to the semantic similarity problem. For this purpose, we present a comparison of the results obtained using our approach in relation to existing methods in the literature.

Algorithm	**Score**	**p-value**
Google distance [16]	0.470	$8.8 \cdot 10^{-3}$
Huang et al. [33]	0.659	$7.5 \cdot 10^{-5}$
Jiang & Conrath [37]	0.669	$5.3 \cdot 10^{-5}$
Resnik [58]	0.780	$1.9 \cdot 10^{-7}$
Leacock & Chodorow [42]	0.807	$4.0 \cdot 10^{-8}$
Lin [44]	0.810	$3.0 \cdot 10^{-8}$
Faruqui & Dyer [22]	0.817	$2.0 \cdot 10^{-8}$
Mikolov et al. [54]	0.820	$2.2 \cdot 10^{-8}$
CoTO [49]	0.850	$1.0 \cdot 10^{-8}$
Our approach	0.855	$1.0 \cdot 10^{-8}$

Table 4: Results from the different algorithms over the Miller & Charles dataset

The first comparison belongs to the domain of the measurement of semantic similarity in general purpose settings. The most successful methods so far are calculated by creating a vectorized representation of the words (a. k. a. word embeddings) using neural networks over large corpora of text [33, 22, 54]. Table 4 shows us the results of different approaches when solving the Miller & Charles dataset. As it can be seen, our approach gets the best results although it is true that the results of neuronal techniques are very dependent on the textual corpus with which they are trained.

Table 5 shows us the results of different approaches when solving the GeReSiD dataset. Unfortunately, this benchmark dataset is not yet very well known, and therefore there are few works that include it. In addition, some authors use the Spearman correlation coefficient that identifies very well what happens inside the benchmark dataset by guessing the relative order, but has little predictive capacity when requiring absolute values. In order to have a neural approach in the pool of compared methods, we have implemented a simple version of LSTM to solve the problem. Once again, our approach is able to overcome the rest of methods.

Table 6 shows the results achieved by the different approaches when solving the MeSH benchmark dataset. The biomedical domain has a long tradition of research in the field of semantic similarity measures, mainly due to the large interoperability problems that exist between different teams of doctors and practitioners from a number of different backgrounds. This means that there are already

Algorithm	**Score**	**p-value**
Han et al. (UMBC) [25]	0.490	$3.0 \cdot 10^{-4}$
Deerwester et al. (LSA) [21]	0.540	$5.2 \cdot 10^{-5}$
Deerwester et al. (LSA2) [21]	0.594	$3.5 \cdot 10^{-7}$
Han et al. (UMBC-STS) [25]	0.630	$4.7 \cdot 10^{-7}$
Aouicha et al. [7]	0.640	$4.7 \cdot 10^{-7}$
O-LSTM (own implementation)	0.675	$2.8 \cdot 10^{-7}$
Our approach	0.729	$1.0 \cdot 10^{-8}$

Table 5: Results from the different algorithms over the GeReSiD dataset

Algorithm	**Score**	**p-value**
Adapted Lesk [9]	0.584	$9.2 \cdot 10^{-4}$
Path-based [57]	0.584	$9.2 \cdot 10^{-4}$
Li et al. [43]	0.707	$7.2 \cdot 10^{-7}$
J&C [37]	0.718	$4.1 \cdot 10^{-7}$
Lin [44]	0.718	$4.1 \cdot 10^{-7}$
Resnik [58]	0.721	$4.0 \cdot 10^{-7}$
Meng et al. [53]	0.731	$2.1 \cdot 10^{-7}$
Seco et al. [61]	0.732	$2.1 \cdot 10^{-7}$
Sanchez et al. [59]	0.735	$1.8 \cdot 10^{-7}$
Taieb et al. [63]	0.753	$6.0 \cdot 10^{-8}$
Our approach	0.774	$2.0 \cdot 10^{-8}$

Table 6: Results from the different algorithms over the MeSH dataset

some methods in this context that work quite well. However, we can see that our FLC is able to perform better in this context.

As a result, we can observe that despite our model is built by a set of restrictions aimed at ensuring interpretability, we have been able to achieve results that are in line with the best methods available. In addition, it is very important to remark that all the ssm that we have chosen as inputs for our semantic similarity controllers are classical algorithms that although they do not yield the best results, are able to operate on manually compiled dictionaries. This makes the whole process highly interpretable. We could always add other ssm based on the use of ANNs that provide very good scores as inputs (e.g. Mikolov et al. [54] in Miller & Charles or Aouicha et al. [7] in MeSH). As a consequence, it could even be possible to improve the results presented in this work. However, this improvement would be achieved at the expense of interpretability. In contrast, the solution that we have presented here has the advantage that it makes it possible to trace the whole process. This means that everything can be easily understood by a human operator without affecting performance, as our initial hypothesis suggested.

6. Conclusions and Future Work

In this work, we have presented our research towards the automatic design of semantic similarity controllers based on fuzzy logics. This research has been guided by the need for a solution capable of expressing the behavior of a sophisticated similarity aggregation strategy in an understandable way. To do that, we have benefited from the use of a particular case of fuzzy systems, i.e. FLCs, which are capable of dealing with imprecise information by encoding expert knowledge directly using fuzzy rules associated with linguistic terms. During the process, we have made use of genetic algorithms techniques to automate the identification phase, and in this way, reduce the time and cost to obtain the final design.

To the best of our knowledge, our solution is the first attempt to face neural approaches in relation to the challenge of measuring semantic similarity from an interpretability perspective and without renouncing to achieve reasonable levels of accuracy. In fact, our experimental evaluation shows some evidence that our approach could be better than methods in the sense that it is able of putting together a non-linear behaviour with an interpretable description in terms of a fully operational IEC 61131-7 program. To reach this conclusion, the proposed approach has been validated on three popular benchmark datasets on semantic similarity measurement with the aim of comparing its performance in terms of accuracy and interpretability with the methods that are known to perform very well in this context.

As future work, we propose to further investigate three research lines. Firstly, it could be of great interest to consider distribution comparison metrics for qualitatively assessing the accuracy of the existing solutions in this field. For example, by measuring of the amount of overlap between the human and the artificially generated results. This could be of particular interest in cases whereby result shows a notable difference especially with many values equal to zero. Secondly, it could be also interesting to further investigate strategies to obtain semantic similarity controllers that are optimized not only to solve Pearson-like problems, but also Spearman-like. In this work, we have focused on optimizing for the Pearson correlation coefficient in order to be able to predict semantic similarity in the future. However, there may be cases where Spearman rank correlation could also make sense, since the user could be interested in ordinal properties of the similarity assessment. Finally, we will work in the further exploration of multi-objective optimization strategies to simultaneously optimize accuracy and interpretability. The proposal introduced in this paper has been able to configure the controller with the minimum number of constraints required to guarantee the best practices on interpretability, but a multi-objective approach could offer the possibility to choose between a wide range of solutions from a Pareto front according to the particular needs of different situations.

Competing interest

Authors have no competing interest to declare.

Acknowledgments

We would like to thank the reviewers for their help towards improving this work. This research work has been partially supported by the Austrian Ministry for Transport, Innovation and Technology, the Federal Ministry of Science, Research and Economy, and the Province of Upper Austria in the frame of the COMET center SCCH.

References

[1] Alcalá, R., Alcalá-Fdez, J., Gacto, M. J., & Herrera, F. (2009). Improving fuzzy logic controllers obtained by experts: a case study in HVAC systems. *Appl. Intell.*, *31*, 15–30. doi:10.1007/

s10489-007-0107-6.

[2] Alcalá-Fdez, J., Alcalá, R., Gacto, M. J., & Herrera, F. (2009). Learning the membership function contexts for mining fuzzy association rules by using genetic algorithms. *Fuzzy Sets and Systems*, *160*, 905–921. doi:10.1016/j.fss.2008.05.012.

[3] Alonso, J. M., & Magdalena, L. (2011). HILK++: an interpretability-guided fuzzy modeling methodology for learning readable and comprehensible fuzzy rule-based classifiers. *Soft Comput.*, *15*, 1959–1980. doi:10.1007/s00500-010-0628-5.

[4] Alonso, J. M., Magdalena, L., & González-Rodríguez, G. (2009). Looking for a good fuzzy system interpretability index: An experimental approach. *Int. J. Approx. Reasoning*, *51*, 115–134. doi:10.1016/j.ijar.2009.09.004.

[5] Angelov, P. P., & Buswell, R. A. (2003). Automatic generation of fuzzy rule-based models from data by genetic algorithms. *Inf. Sci.*, *150*, 17–31. doi:10.1016/S0020-0255(02)00367-5.

[6] Antonelli, M., Ducange, P., Lazzerini, B., & Marcelloni, F. (2009). Multi-objective evolutionary learning of granularity, membership function parameters and rules of mamdani fuzzy systems. *Evolutionary Intelligence*, *2*, 21–37. doi:10.1007/s12065-009-0022-3.

[7] Aouicha, M. B., Taieb, M. A. H., & Hamadou, A. B. (2016). LWCR: multi-layered wikipedia representation for computing word relatedness. *Neurocomputing*, *216*, 816–843. URL: https://doi.org/10.1016/j.neucom.2016.08.045. doi:10.1016/j.neucom.2016.08.045.

[8] Ballatore, A., Bertolotto, M., & Wilson, D. C. (2014). An evaluative baseline for geo-semantic relatedness and similarity. *GeoInformatica*, *18*, 747–767. doi:10.1007/s10707-013-0197-8.

[9] Banerjee, S., & Pedersen, T. (2002). An adapted lesk algorithm for word sense disambiguation using wordnet. In *Computational Linguistics and Intelligent Text Processing, Third International Conference, CICLing 2002, Mexico City, Mexico, February 17-23, 2002, Proceedings* (pp. 136–145). URL: https://doi.org/10.1007/3-540-45715-1_11. doi:10.1007/3-540-45715-1_11.

[10] Bobillo, F., & Straccia, U. (2013). Aggregation operators for fuzzy ontologies. *Appl. Soft Comput.*, *13*, 3816–3830. doi:10.1016/j.asoc.2013.05.008.

[11] Bollegala, D., Matsuo, Y., & Ishizuka, M. (2011). A web search engine-based approach to measure semantic similarity between words. *IEEE Trans. Knowl. Data Eng.*, *23*, 977–990. doi:10.1109/TKDE.2010.172.

[12] Caises, Y., Leyva, E., Muñoz, A. G., & Pérez, R. (2010). A genetic learning of fuzzy relational rules. In *FUZZ-IEEE 2010, IEEE International Conference on Fuzzy Systems, Barcelona, Spain, 18-23 July, 2010, Proceedings* (pp. 1–8). doi:10.1109/FUZZY.2010.5584718.

[13] Casillas, J., Carse, B., & Bull, L. (2007). Fuzzy-xcs: A michigan genetic fuzzy system. *IEEE Trans. Fuzzy Systems*, *15*, 536–550. doi:10.1109/TFUZZ.2007.900904.

[14] Castro, J. L., & Delgado, M. (1996). Fuzzy systems with defuzzification are universal approximators. *IEEE Trans. Systems, Man, and Cybernetics, Part B*, *26*, 149–152. doi:10.1109/3477.484447.

[15] Chaves-González, J. M., & Martinez-Gil, J. (2013). Evolutionary algorithm based on different semantic similarity functions for synonym recognition in the biomedical domain. *Knowl.-Based Syst.*, *37*, 62–69. doi:10.1016/j.knosys.2012.07.005.

[16] Cilibrasi, R., & Vitányi, P. M. B. (2007). The google similarity distance. *IEEE Trans. Knowl. Data Eng.*, *19*, 370–383. URL: https://doi.org/10.1109/TKDE.2007.48. doi:10.1109/TKDE.2007.48.

[17] Cingolani, P., & Alcalá-Fdez, J. (2013). jfuzzylogic: a java library to design fuzzy logic controllers according to the standard for fuzzy control programming. *Int. J. Comput. Intell. Syst.*, *6*, 61–75. doi:10.1080/18756891.2013.818190.

[18] Cordón, O. (2011). A historical review of evolutionary learning methods for mamdani-type fuzzy

rule-based systems: Designing interpretable genetic fuzzy systems. *Int. J. Approx. Reasoning*, *52*, 894–913. doi:10.1016/j.ijar.2011.03.004.

[19] Cordón, O. (2012). A historical review of mamdani-type genetic fuzzy systems. In *Combining Experimentation and Theory - A Hommage to Abe Mamdani* (pp. 73–90). doi:10.1007/978-3-642-24666-1_6.

[20] Cordón, O., Gomide, F. A. C., Herrera, F., Hoffmann, F., & Magdalena, L. (2004). Genetic fuzzy systems. new developments. *Fuzzy Sets and Systems*, *141*, 1–3. doi:10.1016/S0165-0114(03)00110-6.

[21] Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., & Harshman, R. A. (1990). Indexing by latent semantic analysis. *JASIS*, *41*, 391–407.

[22] Faruqui, M., & Dyer, C. (2014). Improving vector space word representations using multilingual correlation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2014, April 26-30, 2014, Gothenburg, Sweden* (pp. 462–471).

[23] Gacto, M. J., Alcalá, R., & Herrera, F. (2011). Interpretability of linguistic fuzzy rule-based systems: An overview of interpretability measures. *Inf. Sci.*, *181*, 4340–4360. doi:10.1016/j.ins.2011.02.021.

[24] Grabisch, M., Marichal, J., Mesiar, R., & Pap, E. (2011). Aggregation functions: Construction methods, conjunctive, disjunctive and mixed classes. *Inf. Sci.*, *181*, 23–43. doi:10.1016/j.ins.2010.08.040.

[25] Han, L., Finin, T., McNamee, P., Joshi, A., & Yesha, Y. (2013). Improving word similarity by augmenting PMI with estimates of word polysemy. *IEEE Trans. Knowl. Data Eng.*, *25*, 1307–1322. doi:10.1109/TKDE.2012.30.

[26] Hellendoorn, H., & Thomas, C. (1993). Defuzzification in fuzzy controllers. *Journal of Intelligent and Fuzzy Systems*, *1*, 109–123. doi:10.3233/IFS-1993-1202.

[27] Hernández, J. A. M., Gomez-Castañeda, F., & Moreno-Cadenas, J. A. (2009). An evolving fuzzy neural network based on the mapping of similarities. *IEEE Trans. Fuzzy Systems*, *17*, 1379–1396. doi:10.1109/TFUZZ.2009.2032364.

[28] Herrera, F., Lozano, M., & Verdegay, J. L. (1995). Tuning fuzzy logic controllers by genetic algorithms. *Int. J. Approx. Reasoning*, *12*, 299–315. doi:10.1016/0888-613X(94)00033-Y.

[29] Ho, W. L., Tung, W. L., & Quek, C. (2010). An evolving mamdani-takagi-sugeno based neural-fuzzy inference system with improved interpretability-accuracy. In *FUZZ-IEEE 2010, IEEE International Conference on Fuzzy Systems, Barcelona, Spain, 18-23 July, 2010, Proceedings* (pp. 1–8). doi:10.1109/FUZZY.2010.5584831.

[30] Holland, J. H., & Reitman, J. S. (1977). Cognitive systems based on adaptive algorithms. *SIGART Newsletter*, *63*, 49. doi:10.1145/1045343.1045373.

[31] Homaifar, A., & McCormick, E. (1995). Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms. *IEEE Trans. Fuzzy Systems*, *3*, 129–139. doi:10.1109/91.388168.

[32] Hsu, H., & Chen, C. (1996). Aggregation of fuzzy opinions under group decision making. *Fuzzy Sets and Systems*, *79*, 279–285. doi:10.1016/0165-0114(95)00185-9.

[33] Huang, E. H., Socher, R., Manning, C. D., & Ng, A. Y. (2012). Improving word representations via global context and multiple word prototypes. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8-14, 2012, Jeju Island, Korea - Volume 1: Long Papers* (pp. 873–882).

[34] IEC-61131-7 (2000). Programmable controllers - part 7: Fuzzy control programming. *Fuzzy Sets and Systems*, .

[35] Ishibuchi, H., Murata, T., & Türksen, I. B. (1997). Single-objective and two-objective genetic al-

gorithms for selecting linguistic rules for pattern classification problems. *Fuzzy Sets and Systems*, *89*, 135–150. doi:10.1016/S0165-0114(96)00098-X.

[36] Jain, R., Sivakumaran, N., & Radhakrishnan, T. K. (2011). Design of self tuning fuzzy controllers for nonlinear systems. *Expert Syst. Appl.*, *38*, 4466–4476. doi:10.1016/j.eswa.2010.09.118.

[37] Jiang, J. J., & Conrath, D. W. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the 10th Research on Computational Linguistics International Conference, ROCLING 1997, Taipei, Taiwan, August 1997* (pp. 19–33).

[38] Kedem, D., Tyree, S., Weinberger, K. Q., Sha, F., & Lanckriet, G. R. G. (2012). Non-linear metric learning. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.* (pp. 2582–2590).

[39] Kiguchi, K., Watanabe, K., & Fukuda, T. (2002). Generation of efficient adjustment strategies for a fuzzy-neuro force controller using genetic algorithms - application to robot force control in an unknown environment. *Inf. Sci.*, *145*, 113–126. doi:10.1016/S0020-0255(02)00226-8.

[40] Lan, W., & Xu, W. (2018). Neural network models for paraphrase identification, semantic textual similarity, natural language inference, and question answering. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018* (pp. 3890–3902).

[41] Lastra-Díaz, J. J., & García-Serrano, A. (2015). A novel family of ic-based similarity measures with a detailed experimental survey on wordnet. *Eng. Appl. of AI*, *46*, 140–153. doi:10.1016/j.engappai.2015.09.006.

[42] Leacock, C., & Chodorow, M. (1998). Combining local context and wordnet similarity for word sense identification. *WordNet: An electronic lexical database*, *49*, 265–283.

[43] Li, Y., Bandar, Z., & McLean, D. (2003). An approach for measuring semantic similarity between words using multiple information sources. *IEEE Trans. Knowl. Data Eng.*, *15*, 871–882. doi:10.1109/TKDE.2003.1209005.

[44] Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998), Madison, Wisconsin, USA, July 24-27, 1998* (pp. 296–304).

[45] Mamdani, E. H., & Assilian, S. (1999). An experiment in linguistic synthesis with a fuzzy logic controller. *Int. J. Hum.-Comput. Stud.*, *51*, 135–147. doi:10.1006/ijhc.1973.0303.

[46] Martinez-Gil, J. (2014). An overview of textual semantic similarity measures based on web intelligence. *Artif. Intell. Rev.*, *42*, 935–943. doi:10.1007/s10462-012-9349-8.

[47] Martinez-Gil, J. (2015). Automated knowledge base management: A survey. *Comput. Sci. Rev.*, *18*, 1–9. URL: https://doi.org/10.1016/j.cosrev.2015.09.001. doi:10.1016/J.COSREV.2015.09.001.

[48] Martinez-Gil, J. (2016). Accurate semantic similarity measurement of biomedical nomenclature by means of fuzzy logic. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, *24*, 291–306. doi:10.1142/S0218488516500148.

[49] Martinez-Gil, J. (2016). Coto: A novel approach for fuzzy aggregation of semantic similarity measures. *Cognitive Systems Research*, *40*, 8–17. doi:10.1016/j.cogsys.2016.01.001.

[50] Martinez-Gil, J. (2019). Semantic similarity aggregators for very short textual expressions: a case study on landmarks and points of interest. *J. Intell. Inf. Syst.*, *53*, 361–380. doi:10.1007/s10844-019-00561-0.

[51] Martinez-Gil, J., & Aldana-Montes, J. F. (2011). Evaluation of two heuristic approaches to solve the ontology meta-matching problem. *Knowl. Inf. Syst.*, *26*, 225–247. URL: https://doi.org/10.1007/s10115-009-0277-0. doi:10.1007/S10115-009-0277-0.

[52] Martinez-Gil, J., & Aldana-Montes, J. F. (2013). Semantic similarity measurement using historical google search patterns. *Inf. Syst. Frontiers*, *15*, 399–410. URL: https://doi.org/10.1007/s10796-012-9404-7. doi:10.1007/S10796-012-9404-7.

[53] Meng, L., Gu, J., & Zhou, Z. (2012). A new model of information content based on concept's topology for measuring semantic similarity in wordnet. *International Journal of Grid and Distributed Computing*, *5*, 81–94.

[54] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.* (pp. 3111–3119).

[55] Miller, G., & Charles, W. (1991). Contextual correlates of semantic similarity. *Language and Cognitive Processes*, *6*, 1–28.

[56] Pedersen, T., Pakhomov, S. V. S., Patwardhan, S., & Chute, C. G. (2007). Measures of semantic similarity and relatedness in the biomedical domain. *Journal of Biomedical Informatics*, *40*, 288–299. doi:10.1016/j.jbi.2006.06.004.

[57] Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes* (pp. 448–453). URL: http://ijcai.org/Proceedings/95-1/Papers/059.pdf.

[58] Resnik, P. (1999). Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *J. Artif. Intell. Res.*, *11*, 95–130. doi:10.1613/jair.514.

[59] Sánchez, D., Batet, M., & Isern, D. (2011). Ontology-based information content computation. *Knowl.-Based Syst.*, *24*, 297–303. URL: `https://doi.org/10.1016/j.knosys.2010.10.001`. doi:`10.1016/j.knosys.2010.10.001`.

[60] Sánchez, L., Couso, I., & Casillas, J. (2009). Genetic learning of fuzzy rules based on low quality data. *Fuzzy Sets and Systems*, *160*, 2524–2552. doi:`10.1016/j.fss.2009.03.004`.

[61] Seco, N., Veale, T., & Hayes, J. (2004). An intrinsic information content metric for semantic similarity in wordnet. In *Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI'2004, including Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004* (pp. 1089–1090).

[62] Smith, S. F. (1980). A learning system based on genetic adaptive algorithms. *PhD thesis, University of Pittsburgh*, .

[63] Taieb, M. A. H., Aouicha, M. B., & Hamadou, A. B. (2014). Ontology-based approach for measuring semantic similarity. *Eng. Appl. of AI*, *36*, 238–261. URL: `https://doi.org/10.1016/j.engappai.2014.07.015`. doi:`10.1016/j.engappai.2014.07.015`.

[64] Xiong, N. (2011). Learning fuzzy rules for similarity assessment in case-based reasoning. *Expert Syst. Appl.*, *38*, 10780–10786. doi:`10.1016/j.eswa.2011.01.151`.