NSS
Neutrosophic Sets and Systems, Vol. 86, 2025
University of New Mexico

Indeterminate and Hierarchical Modeling of Instructional Effectiveness in University Chinese Language and Literature: A Python-Based Approach

Dalun Zhu*, Yongfo Xiong, Qun Xia
Nanchang Institute of Science and Technology, Nanchang, 330108, China
*Corresponding author, E-mail: 474458032@qq.com; rabbit19870228@126.com

Abstract: This paper applies four advanced soft set extensions HyperSoft Set, IndetermSoft Set, IndetermHyperSoft Set, and TreeSoft Set to evaluate teaching quality in university Chinese Language and Literature courses. Each extension is mathematically defined and illustrated through two case studies per model, using realistic data scenarios. A comprehensive practical case study applies all models to a unified dataset, solved step-by-step with numerical data. Python implementations and a results table consolidate the findings, highlighting the models' ability to handle multi-attribute, indeterminate, and hierarchical data in educational assessments.

Keywords: Soft set extensions; teaching quality; university Chinese Language and Literature courses; educational assessments

1. Introduction

Evaluating teaching quality in Chinese Language and Literature courses requires frameworks that manage complex, multi-dimensional, and often uncertain data. Soft set theory, introduced by Molodtsov in 1999 [1].

Recent advancements in soft set theory have led to the introduction of several novel extensions, which broaden the theoretical and practical applications of soft computing models. Among these, the HyperSoft Set, IndetermSoft Set, IndetermHyperSoft Set, SuperHyperSoft Set, TreeSoft Set, and ForestSoft Set were introduced by Florentin Smarandache starting in 2018. These extensions aim to address higher levels of uncertainty, indeterminacy, and hierarchical data representation within soft set environments. Each of these models presents unique structural and functional characteristics that enrich the classical soft set framework and provide robust tools for handling complex decision-making scenarios. For detailed descriptions and foundational work on these soft set extensions, the reader is referred to Smarandache's contributions available at: F. Smarandache, TSS, https://fs.unm.edu/TSS/.

This study has three main contributions:

i. It provides formal mathematical definitions for each model and applies them to realistic case studies relevant to Chinese language teaching.
ii. It presents Python-based implementations to demonstrate the practical feasibility of using these models in institutional settings.
iii. It offers a unified practical case study combining all models on a shared dataset to highlight their comparative strengths.

The paper also acknowledges its limitations, including the reliance on simulated data and the need for empirical validation. Recommendations for future work include the integration of real institutional data, development of hybrid evaluation models (e.g., combining soft sets with fuzzy or neutrosophic logic), and exploration of applications in other academic disciplines such as STEM education.

1.1 A Python Toolkit

To make the paper accessible to readers unfamiliar with soft set theory, we introduce a mathematical framework for handling uncertainty, such as HyperSoft Set and TreeSoft Set, address complex, hierarchical, and indeterminate data, as applied in this study[3].
Key terms:

1) Universe (U): The set of all objects (e.g., lecturers).
2) Power Set ($\mathcal{P}(U)$) : All possible subsets of U.
3) Attributes: Criteria for evaluation (e.g., Teaching Effectiveness).
4) Mapping Function (F) : Assigns attribute values to subsets of U.

2. HyperSoft Set

HyperSoft Set is an extension of soft set theory designed to model complex multi-attribute systems by allowing combinations of parameter values. It was proposed by Florentin Smarandache and builds upon the foundational work of Molodtsov, offering enhanced flexibility in decision-making under uncertainty [2,4].

2.1 Definition

Let U be a universe of discourse, and $\mathcal{P}(U)$ its power set. Let $E = \{e_1, e_2, \dots, e_n\}$ be a set of attributes, with each e_i having a value set A_i, where $|A_i| \geq 1$. Define the attribute combination set as the Cartesian product $A = A_1 \times A_2 \times \cdots \times A_n$. A HyperSoft Set over U is a pair (F, A), where $F: A \to \mathcal{P}(U)$ is a mapping that assigns each attribute combination $a = (a_1, a_2, \dots, a_n) \in A$ to a subset of U.

$$(F, A) = \{(a, F(a)) \mid a \in A, F(a) \subseteq U\}$$

2.2 Example 1: Multi-Attribute Teaching Evaluation

Consider lecturers in a Chinese Language and Literature department, with $U = \{L_1, L_2, L_3, L_4\}$. Attributes are: - a_1 = Teaching Style, with A_1 = { Interactive, Lecture-Based }, $-a_2$ = Content Depth, with A_2 = { Comprehensive, Basic }, $-a_3$ = Student Engagement, with A_3 = { High, Moderate }.
The function F maps attribute combinations to subsets of U, e.g.:

$$F(\text{ Interactive, Comprehensive, High }) = \{L_1, L_3\}.$$

2.3 Python Implementation for Example 1

```
# HyperSoft Set for multi-attribute evaluation
def hypersoft_set(universe, attributes, attribute_values, mapping):
    result = {}
    for attr_comb in mapping:
        result[attr_comb] = mapping[attr_comb]
    return result

# Example data
U = ['L1', 'L2', 'L3', 'L4']
attributes = ['Teaching Style', 'Content Depth', 'Student Engagement']
attribute_values = {
    'Teaching Style': ['Interactive', 'Lecture-Based'],
    'Content Depth': ['Comprehensive', 'Basic'],
    'Student Engagement': ['High', 'Moderate']
}
mapping = {
    ('Interactive', 'Comprehensive', 'High'): ['L1', 'L3'],
    ('Lecture-Based', 'Basic', 'Moderate'): ['L2', 'L4']
}

# Generate the HyperSoft set
hypersoft = hypersoft_set(U, attributes, attribute_values, mapping)

# Display the result
for key, value in hypersoft.items():
    print(f"F{key} = {value}")
```

2.4 Example 2: Course Delivery Preferences

This case study evaluates lecturers based on student preferences for course delivery modes, content relevance, and cultural integration.

Let $U = \{P_1, P_2, P_3, P_4\}$.

Attributes are:

a_1 = Delivery Mode, with A_1 = { Online, In-Person },

a_2 = Content Relevance, with A_2 = { Highly Relevant, Moderately Relevant },

a_3 = Cultural Integration, with A_3 = {Strong, Weak}.

The function $F: A_1 \times A_2 \times A_3 \to P(U)$ has mappings:

$$F(\text{ Online, Highly Relevant, Strong }) = \{P_1, P_2\},$$

$$F(\text{ In-Person, Moderately Relevant, Weak }) = \{P_3, P_4\}.$$

Results: P1 and P2 are best in online delivery with highly relevant content and strong cultural integration, ideal for digital courses. P3 and P4 use in-person teaching with moderately relevant content and weak cultural integration, suitable for traditional courses.

2.5 Python Implementation for Example 2

```python
# HyperSoft Set for course delivery preferences
def hypersoft_set(universe, attributes, attribute_values, mapping):
    result = {}
    for attr_comb in mapping:
        result[attr_comb] = mapping[attr_comb]
    return result

# Example data
U = ['P1', 'P2', 'P3', 'P4']
attributes = ['Delivery Mode', 'Content Relevance', 'Cultural Integration']
attribute_values = {
    'Delivery Mode': ['Online', 'In-Person'],
    'Content Relevance': ['Highly Relevant', 'Moderately Relevant'],
    'Cultural Integration': ['Strong', 'Weak']
}
mapping = {
    ('Online', 'Highly Relevant', 'Strong'): ['P1', 'P2'],
    ('In-Person', 'Moderately Relevant', 'Weak'): ['P3', 'P4']
}

# Generate the HyperSoft set
hypersoft = hypersoft_set(U, attributes, attribute_values, mapping)

# Display the result
for key, value in hypersoft.items():
    print(f"F{key} = {value}")
```

3. IndetermSoft Set

IndetermSoft Set is a theoretical extension of soft set theory developed to handle indeterminate or vague information within single-attribute decision-making. It was introduced by Florentin Smarandache to better represent situations where data is incomplete, ambiguous, or logically uncertain, offering a more nuanced framework for real-world analysis [4].

3.1 Definition

Let U be a universe of discourse, $E = \{e\}$ a single attribute, and A its value set. An IndetermSoft Set is a pair (F, A), where $F: A \to \mathcal{P}(U) \cup \{\langle T, I, F\rangle\}$, and $\langle T, I, F\rangle$ is a neutrosophic set with truth (T), indeterminacy (I), and falsity (F) degrees in [0,1]. The mapping satisfies:

$a \in A$ may have indeterminacy (e.g., $a =$ Other).

$F(a) = \langle T, I, F\rangle$ if indeterminate, where $I > 0$.

$$F(a) = \begin{cases} \{x \subseteq U\} & \text{if determinate,} \\ \langle T, I, F\rangle & \text{if indeterminate, } I > 0 \end{cases}$$

Example: For $U = \{L1, L2\}, A = \{$ Positive, Other $\}, F($ Other $) = \langle 0.3, 0.6, 0.1\rangle$, indicating high indeterminacy.

3.2 Example 1: Uncertain Feedback Analysis

Evaluate teaching quality using student feedback, with $U = \{L_1, L_2, L_3\}$ and $H = \{L_1, L_2\}$. The attribute is $a =$ Feedback Rating, with $A = \{$ Positive, Neutral, Negative, Other $\}$. Indeterminacy arises as:

A includes "Other".
H is uncertain (e.g., " L_3 may be included").
F is indeterminate, e.g., $F($ Positive $) = \{L_1$ or $L_2\}$.

3.3 Python Implementation for Example 1

```
# IndetermSoft Set for uncertain feedback
def indetermsoft_set(universe, subset, attribute_values, indeterminate_mapping):
    result = {}
    for attr in indeterminate_mapping:
        result[attr] = indeterminate_mapping[attr]
    return result

# Example data
U = ['L1', 'L2', 'L3']  # Universe
H = ['L1', 'L2']        # Relevant subset
A = ['Positive', 'Neutral', 'Negative', 'Other']  # Attributes

indeterminate_mapping = {
    'Positive': 'L1 or L2',
    'Negative': 'not L1',
    'Neutral': ['L2'],
    'Other': 'Unknown'
}

# Generate the IndetermSoft set
indetermsoft = indetermsoft_set(U, H, A, indeterminate_mapping)

# Display the result
for key, value in indetermsoft.items():
    print(f"F({key}) = {value}")
```

3.4 Example 2: Ambiguous Student Satisfaction Ratings

This case study assesses lecturers based on student satisfaction ratings, with ambiguity due to vague survey responses. Let $U = \{T_1, T_2, T_3\}$ and $H = \{T_1, T_2\}$.
The attribute is
$a =$ Satisfaction Level, with $A =$ { High, Medium, Low, Unspecified }.
The function $F: A \to P(H)$ has mappings:

$$F(\text{ High }) = \{T_1 \text{ or } T_2\},$$
$$F(\text{ Low }) = \text{ not } T_1,$$
$$F(\text{ Medium }) = \{T_2\},$$
$$F(\text{ Unspecified }) = \text{ Unknown.}$$

Results: High satisfaction is attributed to either T1 or T2. T1 avoids low ratings, while T2 has medium satisfaction. Unspecified ratings reflect data gaps, prompting improved survey design.

3.5 Python Implementation for Case Study 2

```python
# IndetermSoft Set for ambiguous satisfaction ratings
def indetermsoft_set(universe, subset, attribute_values, indeterminate_mapping):
    result = {}
    for attr in indeterminate_mapping:
        result[attr.strip()] = indeterminate_mapping[attr]
    return result

# Example data
U = ['T1', 'T2', 'T3']  # Universe
H = ['T1', 'T2']        # Subset (relevant elements)
A = ['High', 'Medium', 'Low', 'Unspecified']  # Attribute values

indeterminate_mapping = {
    'High': 'T1 or T2',
    'Low': 'not T1',
    'Medium': ['T2'],
    'Unspecified': 'Unknown'
}

# Generate the IndetermSoft set
indetermsoft = indetermsoft_set(U, H, A, indeterminate_mapping)

# Display the result
for key, value in indetermsoft.items():
    print(f"F({key}) = {value}")
```

4. IndetermHyperSoft Set

IndetermHyperSoft Set is a hybrid extension of soft set theory introduced by Florentin Smarandache. It combines the features of HyperSoft Sets (multi-attribute combinations) with the capacity of IndetermSoft Sets to handle uncertainty and vagueness. This model is particularly useful in scenarios involving complex, uncertain relationships between multiple interdependent attributes.

4.1 Definition

Let U be a universe of discourse, and $E = \{e_1, \ldots, e_n\}$ a set of attributes with value sets A_i. Let $A = A_1 \times \cdots \times A_n$. An IndetermHyperSoft Set is a pair (F, A), where $F: A \to \mathcal{P}(U) \cup \{$ Indeterminate $\}$, and at least one of:

1. Some $a_i \in A_i$ is indeterminate.
2. $F(a)$ is indeterminate.
3. The mapping $F(a)$ is uncertain.

$$(F, A) = \{(a, F(a)) \mid a \in A, F(a) \subseteq U \text{ or } F(a) \text{ is indeterminate } \}$$

4.2 Example 1: Multi-Attribute Uncertain Evaluation

Evaluate teaching quality with $U = \{L_1, L_2, L_3, L_4\}$ and $H = \{L_1, L_2, L_3\}$. Attributes are: a_1 = Teaching Method, with $A_1 = \{$Interactive, Traditional, Other$\}$. a_2 = Assessment Fairness, with $A_2 = \{$ Fair, Unfair $\}$.
Indeterminacy includes "Other" in A_1, uncertain H, and mappings like:

$$F(\text{ Interactive , Fair }) = \{L_1 \text{ or } L_2\}.$$

4.3 Python Implementation for Example 1

```
# IndetermHyperSoft Set for uncertain multi-attribute evaluation
def indetermhypersoft_set(universe, subset, attributes, attribute_values, indeterminate_mapping):
    result = {}
    for attr_comb in indeterminate_mapping:
        result[attr_comb] = indeterminate_mapping[attr_comb]
    return result

# Example data
U = ['L1', 'L2', 'L3', 'L4']
H = ['L1', 'L2', 'L3']
attributes = ['Teaching Method', 'Assessment Fairness']
attribute_values = {
    'Teaching Method': ['Interactive', 'Traditional', 'Other'],
    'Assessment Fairness': ['Fair', 'Unfair']
}
indeterminate_mapping = {
    ('Interactive', 'Fair'): 'L1 or L2',
    ('Traditional', 'Unfair'): 'not L3',
    ('Other', 'Fair'): 'Unknown'
}

# Generate the IndetermHyperSoft set
indetermhypersoft = indetermhypersoft_set(U, H, attributes, attribute_values, indeterminate_mapping)

# Display the result
for key, value in indetermhypersoft.items():
    print(f"F{key} = {value}")
```

4.4 Example 2: Uncertain Faculty Peer Reviews

This case study assesses lecturers based on peer reviews of teaching innovation and student support.

Let $U = \{D_1, D_2, D_3, D_4\}$ and $H = \{D_1, D_2, D_3\}$.

Attributes are:

a_1 = Teaching Innovation, with $A_1 = \{$ Innovative, Conventional, Undefined $\}$.

a_2 = Student Support, with $A_2 = \{$ Excellent, Adequate $\}$.

The function $F: A_1 \times A_2 \to P(H)$ has mappings:

$$F(\text{ Innovative, Excellent }) = \{D_1 \text{ or } D_2\}$$
$$F(\text{ Conventional, Adequate }) = \text{ not } D_3$$
$$F(\text{ Undefined, Excellent }) = \text{ Unknown.}$$

Results: D1 or D2 is innovative with excellent support. D3 avoids conventional teaching with adequate support. Undefined styles yield no clear outcome, highlighting the need for standardized reviews.

4.5 Python Implementation for Example 2

```
# IndetermHyperSoft Set for uncertain peer reviews
def indetermhypersoft_set(universe, subset, attributes, attribute_values, indeterminate_mapping):
    result = {}
    for attr_comb in indeterminate_mapping:
        result[attr_comb] = indeterminate_mapping[attr_comb]
    return result

# Example data
U = ['D1', 'D2', 'D3', 'D4']  # Universe
H = ['D1', 'D2', 'D3']        # Relevant subset
attributes = ['Teaching Innovation', 'Student Support']  # Attributes

attribute_values = {
    'Teaching Innovation': ['Innovative', 'Conventional', 'Undefined'],
    'Student Support': ['Excellent', 'Adequate']
}

indeterminate_mapping = {
    ('Innovative', 'Excellent'): 'D1 or D2',
    ('Conventional', 'Adequate'): 'not D3',
    ('Undefined', 'Excellent'): 'Unknown'
}

# Generate the IndetermHyperSoft set
indetermhypersoft = indetermhypersoft_set(U, H, attributes, attribute_values, indeterminate_mapping)

# Display the result
for key, value in indetermhypersoft.items():
    print(f"F{key} = {value}")
```

5. TreeSoft Set

TreeSoft Set is an advanced extension of soft set theory introduced by Florentin Smarandache to model hierarchical and structured data. Unlike traditional flat attribute systems, TreeSoft Sets represent attributes in tree-like layers such as main attributes, sub-attributes, and sub-sub-attributes making them ideal for analyzing systems with nested or multi-level criteria, such as educational assessment or organizational decision-making.

5.1 Definition

Let U be a universe, and T a graph-tree with first-level attributes $E = \{e_1, \dots, e_n\}$, each with sub-attributes up to level k. Let A be all nodes, and $w: A \to [0,1]$ a weight function. A TreeSoft Set is a pair (F, A), where $F: A \to \mathcal{P}(U)$, and the decision score for $x \in U$ is:

$$\text{Score}(x) = \sum_{a \in A : x \in F(a)} w(a)$$

Example: For $U = \{L1, L2\}$, attributes Pedagogy ($w = 0.6$) and Interaction ($w = 0.4$), sub-attribute Delivery $(w = 0.3)$, $F($ Pedagogy, Delivery $) = \{L1\}$, $\text{Score}(L1) = 0.3$.

5.2 Example 1: Hierarchical Teaching Quality Assessment

Consider $U = \{L_1, L_2, \dots, L_8\}$ and $H = \{L_1, L_2, \dots, L_6\}$.
The attribute set is $A = \{A_1, A_2\}$
A_1 = Pedagogy, with $A_1 = \{A_{1,1}, A_{1,2}\}$ = { Delivery, Content }.
A_2 = Interaction, with $A_2 = \{A_{2,1}\}$ = { Engagement }.

Sub-attributes are:
$A_{1,1}$ = { Clear, engaging }.
$A_{1,2}$ = { Relevant, outdated }.
Example mappings include:

$$F(\text{ Pedagogy , Delivery , Clear }) = \{L_1, L_3\}$$

5.3 Python Implementation for Example 1

```python
# TreeSoft Set for hierarchical evaluation
def treesoft_set(universe, subset, tree_structure, mapping):
    result = {}
    for node in mapping:
        result[node] = mapping[node]
    return result

# Example data
U = ['L1', 'L2', 'L3', 'L4', 'L5', 'L6', 'L7', 'L8']
H = ['L1', 'L2', 'L3', 'L4', 'L5', 'L6']

tree_structure = {
    'Pedagogy': {
        'Delivery': ['Clear', 'Engaging'],
        'Content': ['Relevant', 'Outdated']
    },
    'Interaction': {
        'Engagement': []
    }
}

mapping = {
    ('Pedagogy', 'Delivery', 'Clear'): ['L1', 'L3'],
    ('Pedagogy', 'Content', 'Relevant'): ['L2', 'L4', 'L5'],
    ('Interaction', 'Engagement'): ['L1', 'L2']
}

# Generate the TreeSoft set
treesoft = treesoft_set(U, H, tree_structure, mapping)

# Display the result
for key, value in treesoft.items():
    print(f"F{key} = {value}")
```

5.4 Example 2: Hierarchical Course Evaluation Criteria

This case study evaluates lecturers using a hierarchical framework of teaching effectiveness. Let $U = \{S_1, S_2, \ldots, S_8\}$ and $H = \{S_1, S_2, \ldots, S_6\}$. The attribute set is $A = \{A_1, A_2\}$: $-A_1$ = Preparation, with $A_{1,1}$ = Syllabus Clarity, $A_{1,2}$ = Material Organization. - A_2 = Classroom Dynamics, with $A_{2,1}$ = Discussion Facilitation.
Sub-attributes are: - $A_{1,1}$ = { Clear, Vague }. $-A_{1,2}$ = { Well-Organized, Disorganized }.
The function $F: P(\text{Tree}(A)) \to P(H)$ has mappings:

$$F(\text{ Preparation, Syllabus Clarity, Clear }) = \{S_1, S_3\},$$
$$F(\text{ Preparation, Material Organization, Well-Organized }) = \{S_2, S_4, S_5\},$$
$$F(\text{ Classroom Dynamics, Discussion Facilitation }) = \{S_1, S_6\}.$$

Results: S1 and S3 provide clear syllabi, S2, S4, and S5 offer well-organized materials, and S1 and S6 excel in facilitating discussions.

5.5 Python Implementation for Example 2

```python
# TreeSoft Set for hierarchical course evaluation
def treesoft_set(universe, subset, tree_structure, mapping):
    result = {}
    for node in mapping:
        result[node] = mapping[node]
    return result

# Example data
U = ['S1', 'S2', 'S3', 'S4', 'S5', 'S6', 'S7', 'S8']
H = ['S1', 'S2', 'S3', 'S4', 'S5', 'S6']

tree_structure = {
    'Preparation': {
        'Syllabus Clarity': ['Clear', 'Vague'],
        'Material Organization': ['Well-Organized', 'Disorganized']
    },
    'Classroom Dynamics': {
        'Discussion Facilitation': []
    }
}

mapping = {
    ('Preparation', 'Syllabus Clarity', 'Clear'): ['S1', 'S3'],
    ('Preparation', 'Material Organization', 'Well-Organized'): ['S2', 'S4', 'S5']
}

# Generate the TreeSoft set
treesoft = treesoft_set(U, H, tree_structure, mapping)

# Display the result
for key, value in treesoft.items():
    print(f"F{key} = {value}")
```

5.6 Data Generation Methodology

The simulated dataset for the practical case study was generated using constrained random sampling to mimic real-world teaching evaluation scenarios. For five lecturers (Q1 to $Q5$):

- Teaching Effectiveness: Scores (0-10) drawn from a normal distribution ($\mu = 7, \sigma = 1.5$), categorized as High (≥ 8), Medium (5-7), or Uncertain (missing or ambiguous).
- Assessment Clarity: Categorical values (Clear, Unclear, Unspecified) assigned with probabilities 0.5, 0.3, 0.2, respectively.
- Course Engagement: Hierarchical attributes (Student Participation, Content Interaction) sampled with equal probabilities for Active/Passive and Dynamic/Static.

This methodology ensures realistic variability while maintaining control over uncertainty levels.

6. Practical Case Study Across All Models

This section presents a comprehensive case study applying four advanced soft set extensions HyperSoft Set, IndetermSoft Set, IndetermHyperSoft Set, and TreeSoft Set to evaluate the teaching quality of five lecturers (Q1 to Q5) in a university Chinese Language and Literature department. The assessment draws on simulated data mimicking common sources such as student ratings, peer evaluations, and administrative observations. The study focuses on three main attributes: Teaching Effectiveness, Assessment Clarity, and Course Engagement. Teaching Effectiveness is derived from a numerical scale (0 to 10) and categorized as High (≥ 8), Medium (5–7), or Uncertain. Assessment Clarity is treated categorically as Clear, Unclear, or Unspecified. Course Engagement is modeled hierarchically through two dimensions: Student Participation (Active or Passive) and Content Interaction (Dynamic or Static).

The data set for this case study includes five profiles:

1) Q1: Effectiveness = 8.5 (High), Clarity = Clear, Engagement = Active + Dynamic
2) Q2: Effectiveness = 7.2 or 6.8 (Uncertain), Clarity = Unclear, Engagement = Passive + Static
3) Q3: Effectiveness = 9.0 (High), Clarity = Clear, Engagement = Active + Dynamic
4) Q4: Effectiveness = 6.8 (Medium), Clarity = Unspecified, Engagement = Passive + Static
5) Q5: Effectiveness = Unknown (Uncertain), Clarity = Unspecified, Engagement = Active + Static

6.1 HyperSoft Set Application

The HyperSoft Set model is used to evaluate teaching quality based on crisp, multi-attribute combinations. The three attributes considered are Teaching Effectiveness (High, Medium), Assessment Clarity (Clear, Unclear), and Course Engagement (High, Low). Here, we interpret "High Engagement" as a combination of active participation and dynamic interaction, while "Low Engagement" includes passive or static involvement. Based on this categorization, Q1 and Q3 fall under the tuple (High, Clear, High), whereas Q2 and Q4 are categorized as (Medium, Unclear, Low).

The corresponding Python implementation is shown below:

```
def hypersoft_set(universe, attributes, attribute_values, mapping):
    result = {}
    for attr_comb in mapping:
        result[attr_comb] = mapping[attr_comb]
    return result

U = ['Q1', 'Q2', 'Q3', 'Q4', 'Q5']
attributes = ['Teaching Effectiveness', 'Assessment Clarity', 'Course Engagement']
attribute_values = {
    'Teaching Effectiveness': ['High', 'Medium'],
    'Assessment Clarity': ['Clear', 'Unclear'],
    'Course Engagement': ['High', 'Low']
}
mapping = {
    ('High', 'Clear', 'High'): ['Q1', 'Q3'],
    ('Medium', 'Unclear', 'Low'): ['Q2', 'Q4']
}

hypersoft = hypersoft_set(U, attributes, attribute_values, mapping)
for key, value in hypersoft.items():
    print(f"F{key} = {value}")
```

6.2 IndetermSoft Set Application

The IndetermSoft Set is applied to capture uncertainty in the single attribute of Teaching Effectiveness. Here, Q1 and Q3 are categorized as *High*, Q4 as *Medium*, and Q2 and Q5 are grouped under *Uncertain* due to incomplete or ambiguous ratings. The mapping avoids any instance of *Low* effectiveness. The implementation is concise and clearly separates determinate and indeterminate cases:

```
def indetermsoft_set(universe, attribute_values, mapping):
    return mapping

U = ['Q1', 'Q2', 'Q3', 'Q4', 'Q5']
attribute_values = ['High', 'Medium', 'Low', 'Uncertain']
mapping = {
    'High': ['Q1', 'Q3'],
    'Medium': ['Q4'],
    'Uncertain': ['Q2', 'Q5'],
    'Low': []
}

indetermsoft = indetermsoft_set(U, attribute_values, mapping)
for key, value in indetermsoft.items():
    print(f"F({key}) = {value}")
```

6.3 IndetermHyperSoft Set Application

The IndetermHyperSoft Set is particularly useful when both attribute values and their relationships are uncertain. For this case, the attributes Teaching Effectiveness and Assessment Clarity are considered. Q1 and Q3 exhibit determinate, high-quality profiles (High, Clear). Q2, which contains uncertainty in effectiveness, is mapped to (Uncertain, Unclear), while Q4 fits under (Medium, Unspecified). Q5 is the most ambiguous case, assigned to (Uncertain, Unspecified). The implementation is as follows:

```python
def indetermhypersoft_set(universe, attributes, attribute_values, mapping):
    return mapping

U = ['Q1', 'Q2', 'Q3', 'Q4', 'Q5']
attributes = ['Teaching Effectiveness', 'Assessment Clarity']
attribute_values = {
    'Teaching Effectiveness': ['High', 'Medium', 'Uncertain'],
    'Assessment Clarity': ['Clear', 'Unclear', 'Unspecified']
}
mapping = {
    ('High', 'Clear'): ['Q1', 'Q3'],
    ('Uncertain', 'Unclear'): ['Q2'],
    ('Medium', 'Unspecified'): ['Q4'],
    ('Uncertain', 'Unspecified'): ['Q5']
}

indetermhypersoft = indetermhypersoft_set(U, attributes, attribute_values, mapping)
for key, value in indetermhypersoft.items():
    print(f"F{key} = {value}")
```

6.4 TreeSoft Set Application

Finally, the TreeSoft Set framework is utilized to model the hierarchical structure of course engagement. This includes two sub-branches: *Student Participation* (Active vs. Passive) and *Content Interaction* (Dynamic vs. Static). According to the data, Q1 and Q3 are strong performers with active participation and dynamic content delivery, while Q2 and Q4 demonstrate lower engagement. Q5 shows mixed results, being active yet static. The code is designed to reflect this layered structure:

```python
def treesoft_set(universe, tree_structure, mapping):
    return mapping

U = ['Q1', 'Q2', 'Q3', 'Q4', 'Q5']
tree_structure = {
    'Course Engagement': {
        'Student Participation': ['Active', 'Passive'],
        'Content Interaction': ['Dynamic', 'Static']
    }
}
mapping = {
    ('Student Participation', 'Active'): ['Q1', 'Q3', 'Q5'],
    ('Student Participation', 'Passive'): ['Q2', 'Q4'],
    ('Content Interaction', 'Dynamic'): ['Q1', 'Q3'],
    ('Content Interaction', 'Static'): ['Q2', 'Q4', 'Q5']
}

treesoft = treesoft_set(U, tree_structure, mapping)
for key, value in treesoft.items():
    print(f"F{key} = {value}")
```

6.5 Results Summary

The application of the four soft set extensions HyperSoft Set, IndetermSoft Set, IndetermHyperSoft Set, and TreeSoft Set produced a layered evaluation of teaching quality across the five lecturer profiles analyzed. Each model contributed a distinct perspective on the data, enabling both overlap and divergence in results to emerge.

The HyperSoft Set successfully classified lecturers Q1 and Q3 as high performers based on clear and direct attribute combinations, notably their high effectiveness, clarity in assessment, and strong student engagement. Meanwhile, Q2 and Q4 were consistently grouped under moderate performance profiles, showing lower engagement and less clarity in assessment design.

With the IndetermSoft Set, uncertainty in data came to the forefront. While Q1 and Q3 maintained their high standing, Q2 and Q5 fell into the "uncertain" category due to either ambiguous scores or missing evaluations. This model highlighted the critical need for more precise and complete data collection practices.

The IndetermHyperSoft Set refined this further by exposing composite uncertainty in multi-attribute relationships. It confirmed the strength of Q1 and Q3 but added nuance in categorizing Q2, Q4, and Q5, emphasizing how combined uncertainties in effectiveness and clarity can obscure reliable evaluation.

Finally, the TreeSoft Set illuminated patterns in teaching engagement by breaking down hierarchical behaviors. Lecturers Q1 and Q3 again demonstrated active participation and dynamic classroom interaction, reinforcing their consistency across all models. In contrast, Q2 and Q4 were flagged for passive and static engagement, and Q5 showed mixed signals of active participation yet static content delivery.

The results across all models were convergent in identifying Q1 and Q3 as strong teaching profiles, and Q2 and Q4 as moderately effective but requiring targeted professional development. Q5 remains inconclusive, with substantial gaps in clarity and consistency. This multidimensional approach not only strengthens the reliability of teaching assessments but also underscores the importance of accounting for both structured data and inherent uncertainties in educational environments (see Table 1) .

Table 1: Enhanced Summary of Results

Model	**Attribute Combination**	**Result (U)**
HyperSoft Set	(High, Clear, High)	{ Q1, Q3 }: High-performing lecturers
IndetermSoft Set	High	{ Q1, Q3 }: Consistent excellence
IndetermHyperSoft Set	(High, Clear)	{ Q1, Q3 }: Robust profiles
TreeSoft Set	(Course Engagement, Student Participation, Active)	{ Q1, Q3 }: Dynamic teaching

6.6 Comparative Analysis of Soft Set Extensions

Each soft set extension addresses distinct aspects of teaching quality evaluation:

1) HyperSoft Set: Suitable for crisp, multi-attribute combinations. Complexity: $O(|A| \cdot |U|)$, where $|A|$ is the number of attribute combinations.
2) IndetermSoft Set: Handles single-attribute uncertainty using neutrosophic logic. Complexity: $O(|A| \cdot |U|)$.
3) IndetermHyperSoft Set: Combines multi-attribute and indeterminacy, ideal for complex uncertainty. Complexity: $O(|A| \cdot |U|)$.
4) TreeSoft Set: Models hierarchical data with weighted attributes. Complexity: $O(|A| \cdot |U| + |T|)$, where $|T|$ is the tree size.

HyperSoft Set is simplest but less flexible for uncertainty. IndetermHyperSoft Set is most versatile but computationally intensive. TreeSoft Set excels in structured environments.

6.7 Sensitivity Analysis

To assess model robustness, we varied Teaching Effectiveness scores by $\pm 10\%$ and reevaluated mappings. For HyperSoft Set, F (High, Clear, High) $= \{Q1, Q3\}$ remained stable unless scores dropped below 7.2. IndetermSoft Set was sensitive to changes in Uncertain classifications, suggesting the need for stricter indeterminacy thresholds.

$$\text{New Score } (x) = \text{ Original Score } (x) \cdot (1 \pm 0.1)$$

7. Conclusion

This study demonstrates that advanced soft set models offer a flexible and practical approach to evaluating teaching quality, especially in fields like language and literature where data can be complex or uncertain. By incorporating models that handle ambiguity, multiple attributes, and hierarchical structures, the evaluation process becomes more insightful and fairer. These models are not meant to replace traditional methods, but rather to enhance them. Moving forward, the key to their effectiveness lies in applying them to real-world data and integrating them into institutional assessment practices.

8. Limitations

While this study offers a structured and innovative approach to evaluating teaching quality, it has several limitations. First, the data used is entirely simulated, which means the results may not fully reflect the complexity of real academic environments. Second, the attribute selection and categorization were subjective, which could influence the outcomes. Additionally, some of the models particularly those involving indeterminacy require clearer criteria for handling ambiguous cases.

Finally, the practical implementation relies on basic Python scripts, which may need refinement for use in institutional systems. Future work should focus on

1) Conduct a pilot study with real teaching evaluation data from a Chinese university.
2) Develop a machine learning model to predict teaching quality using soft set outputs.
3) Apply the framework to STEM education to test generalizability.

4) Create an open-source Python package for soft set evaluations.

Acknowledgment
The research project of teaching reform in Jiangxi Province's colleges and universities, "Teaching reform and practical innovation of the" five integration "course under the golden curriculum goal" (No.: JXJG-21-27-1).

Data Availability
The datasets used in this study are simulated for demonstration purposes and were designed to reflect realistic teaching evaluation scenarios. While they are not based on actual institutional data, they are available upon reasonable requests from the corresponding author.

Code Availability
The Python code implementations for all soft set models used in this study have been developed specifically for educational and research use. Researchers interested in accessing the source code may contact the corresponding author to request a copy.

References
[1] Molodtsov, Dmitriy. "Soft set theory—first results." Computers & mathematics with applications 37.4-5 (1999): 19-31.
[2] Smarandache, F. (2018). Extension of soft set to hypersoft set, and then to plithogenic hypersoft set. Neutrosophic Sets and Systems: An International Book Series in Information Science and Engineering, vol. 22/2018, 22, 168.
[3] Smarandache, F. (2022). Introduction to the IndetermSoft Set and IndetermHyperSoft Set (Vol. 1). Infinite Study.
[4] Smarandache, F. (2023). New types of soft sets "hypersoft set, indetermsoft set, indetermhypersoft set, and treesoft set": an improved version. Infinite Study.

Received: Nov. 19, 2024. Accepted: May 18, 2025