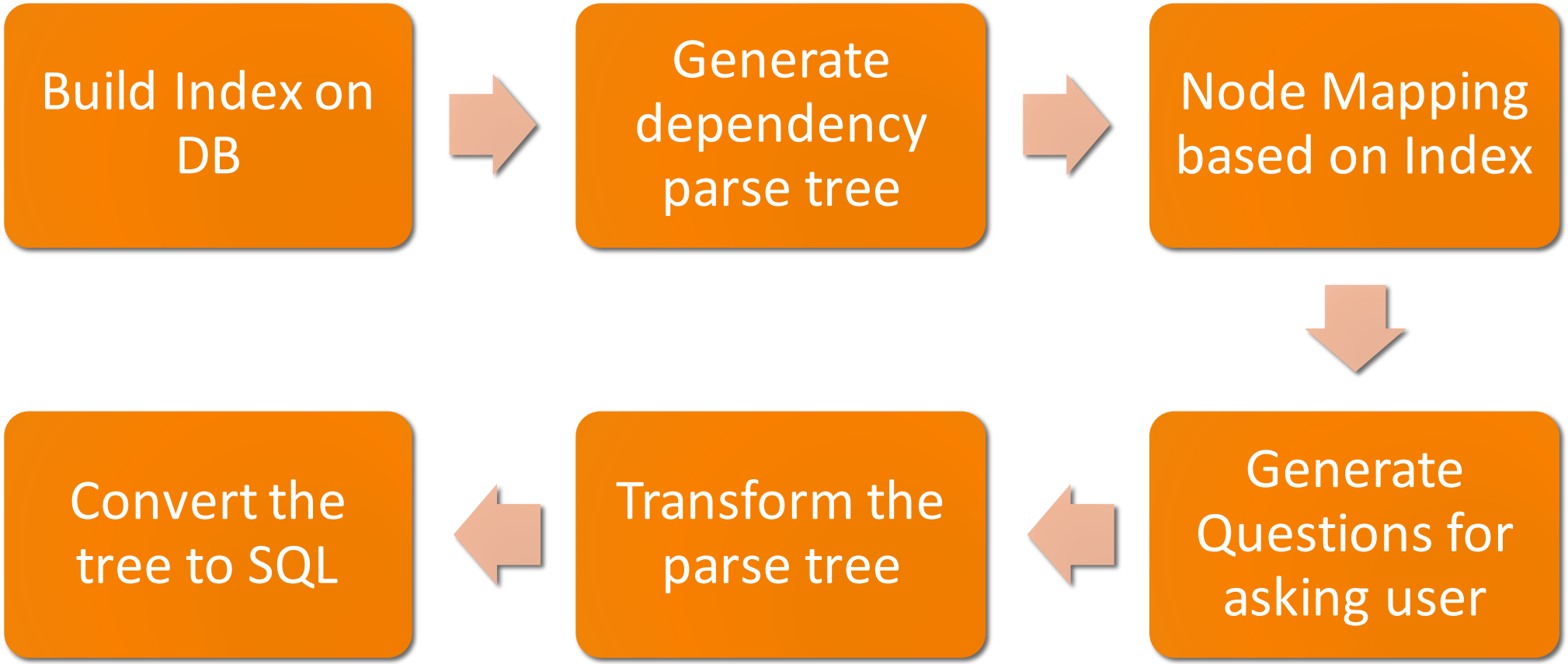
# NILE: An Interactive Natural Language Interface for Relational Databases

ANIL SHANBHAG (ANILS@MIT.EDU)

# Problem

Given mapping of {natural language query => SQL} on specific domains, we want to get the correct SQL given natural language queries on an unseen domain.

# Approach


Build Index on DB
Generate dependency parse tree
Node Mapping based on Index
Generate Questions for asking user
Transform the parse tree
Convert the tree to SQL

# Node Mapping

Implemented as phases:

Annotate Select Node (based on keywords – where,which,how,..)

Annotate Database Entitites

- Uses index to lookup token
- Uses nltk.lemmatize for plurals (NNS/NNPS) Eg: cities -> city
- In case of multiple matches, looks at modifiers. If unable to reduce number of candidates to 1, asks user. Eg: "What are the high points of states surrounding mississippi ?"

  Mississippi can be a state or river

Deletes nodes which have no effect (PRP, DT, VB's with no children)

# QA Phase

After the node mapping phase, all the nodes which have not been previously annotated would be annotated.

We want to annotate all the VB*(verbs), NN*(nouns) and JJ*(adjectives)

For eg:

What is population density of texas ?

Density has context state and state.population. *Expr:* state.population / state.area

What are the high points of states surrounding texas?

Surrounding has context state. *Expr:* border.state_id; border.border_state_id = state.id

Currently requires knowledge but maybe these can be given as natural language (future work)

# Transform

Approach #1: Brute

Brute has transformations hard coded.

Eg:

- Moving the select node to the root
- Moving the parent of preposition in select clause if any to where clause
- Merge with parent in case of compound

Each of these run as a pass on the tree.

# Transform

Approach #2: Logistic Regression

For every node in the tree which is not fitting grammar, we want to do one of the actions:

- Swap with parent
- Combine with parent
- Become sibling of parent

These basic transformations allow us to go from any tree to any other tree.

Using the train set, we train a model which given the context predicts the action to be taken.

Features used are:

[POS Tag, arc tag, annotation] of each of parent, leftmost, rightmost and node itself.

# Convert to SQL

The transformation ensure that tree root is select node. The first child is the select clause and the rest of the children constitute the predicates.

First we connect to the database to get the entire db schema, this is used in many of the steps.

From the schema, we construct a schema graph.

For each predicate, we construct the predicate based on expr / value node. Then we use the schema graph to make this filter apply on the target table. (simple BFS).

# Full Example

What are all the rivers in colorado ?

Initial tree: what WP ROOT SN ( are VBP cop D , rivers NNS nsubj NN( all PDT det:predet UD , the DT det D , colorado NN nmod VN( in IN case UD ) ) )

What is colorado ? Options: [['colorado', Column(name=name, table_name=state)], ['colorado', Column(name=name, table_name=river)], ['colorado springs', Column(name=name, table_name=city)], ['colorado river', Column(name=lowest_point, table_name=highlow)] ? 0

What is “all” in terms of Table(name=river, columns=[…]) ? River

After filling gaps: what WP ROOT SN( are VBP cop D , rivers NNS nsubj NN( all PDT det:predet D , the DT det D , colorado NN nmod VN( in IN case UD ) ) )

Removed deleted: what WP ROOT SN( rivers NNS nsubj NN( colorado NN nmod VN( in IN case UD ) ) )

Converted tree: what WP ROOT SN( rivers NNS nsubj NN , colorado NN nmod VN( in IN case UD ) )

SELECT river.* FROM river WHERE river.id = (SELECT flows_through.river_id FROM state INNER JOIN flows_through ON flows_through.state_id = state.id WHERE state.name = colorado)

# Results

Test Set: 379 queries from the Geo880 dataset

Train Set: 30 queries from Jobs640 dataset manually converted

| Method | Precision | Recall |
|---|---|---|
| Approach 1: Brute | 98.22 | 60.4 |
| Approach 2: LR | 100 | 31.13 |
| Zettlemoyer and Collins | 96.25 | 79.29 |

# ToDo

Construct schema for Jobs dataset.

Use it to run brute to collect a larger training dataset for the LR approach

Improve the SQL generator

Handle aggregation