Developing a Machine Learning Approach for Schedule Summarization

Daniel Newkirk, PhD

Discussion Points

- Introduction
- Natural Language Processing (NLP)
- Hidden Markov Models (HMM)
- Design Considerations
- Results

Introduction

Perform iSRA or JCL

Summarization is difficult and time intensive

Build by hand

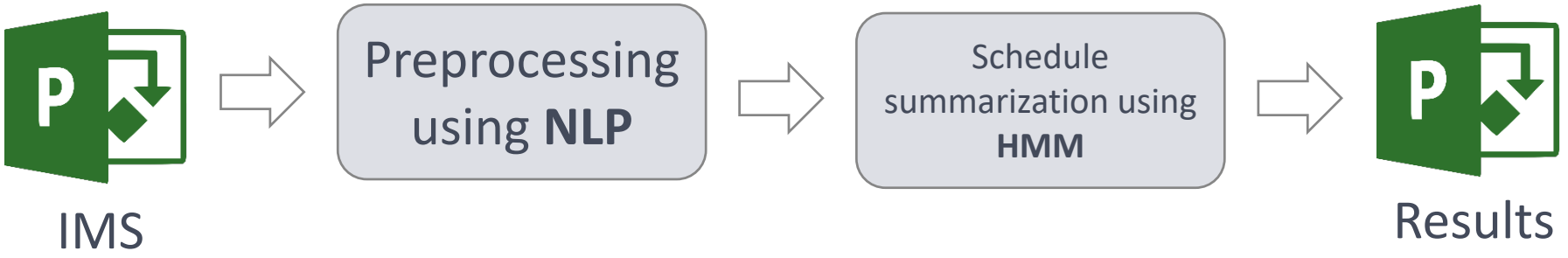
Premise

Machine Learning can be used speed up this process and suggest the final categories.

Benefits of Machine Learning

- Consistency
- Prevent Mistakes
- Increased Speed

Schedule Summarization Approach



Definitions

- IMS = Integrated Master Schedule
- NLP = Natural Language Processing
- HMM = Hidden Markov Models
- Results = Analysis Schedule

IMS Challenges for a Large Space Program

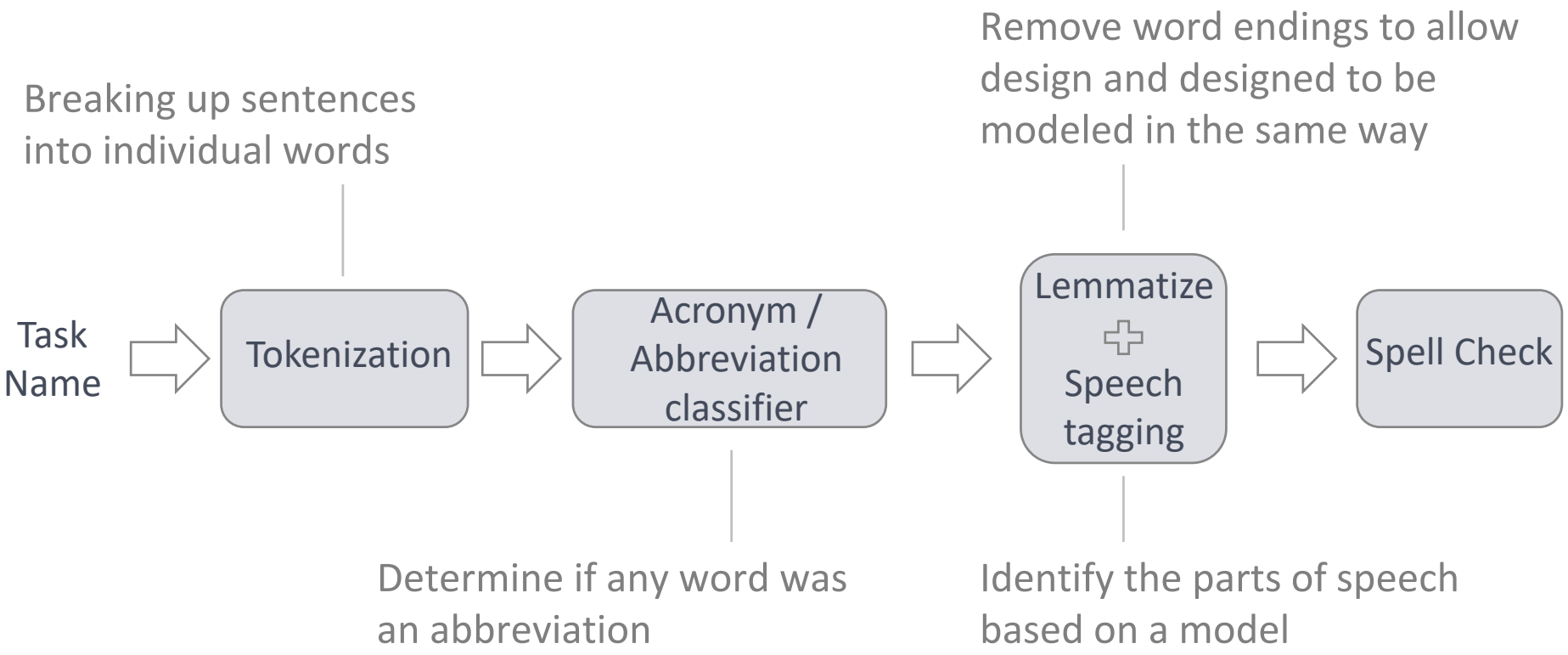
An IMS can have…

- Overlapping meta tasks
  - Particularly during acquisition/development, meta tasks in the summary schedule are likely to overlap in time
- Variable text-based descriptions
  - Can use two separate sentences/descriptions for the same task performed for two separate space vehicles in the same schedule
- Spelling errors
  - "Testng" instead of "testing"
- Abbreviations/acronyms
  - TVAC
- Concatenated words (scheduler forgot to add a space between words)
  - "Testingcomplete" instead of "testing complete"

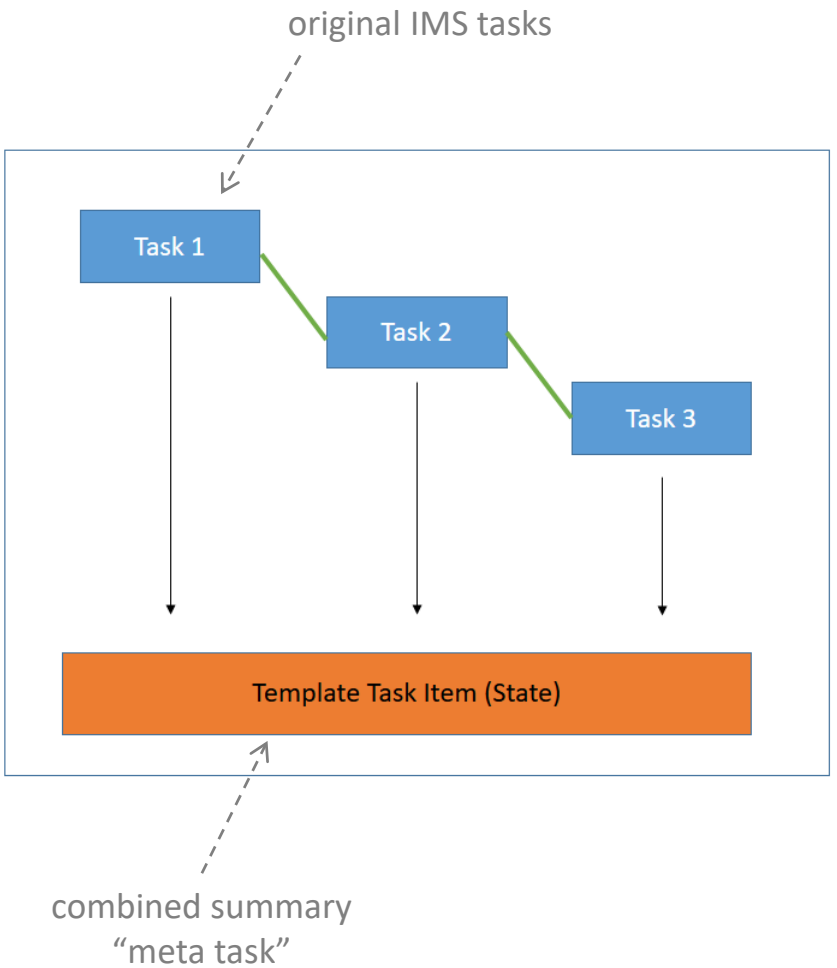
Primer: Natural Language Processing

- A broad discipline that evaluates the how a computer is able to utilize human language in decision making
  - Machine Learning, Artificial Intelligence
- Some different tools in NLP:
  - Bag of words (words as independent, word frequency)
  - N-gram analysis (words adjacent to one another, i.e. bigram, etc.)
  - **Word Set/combinations (unordered word combinations)**
  - Statistical parsing (evaluating grammar and intended meaning in a sentence)
  - Sentiment analysis (is the author treating the topic negatively?)
  - Author identification (who is the author, and what do we know about them-socioeconomic status, religion, ethnicity, etc.)
- Example: Spam filter

NLP Process Workflow

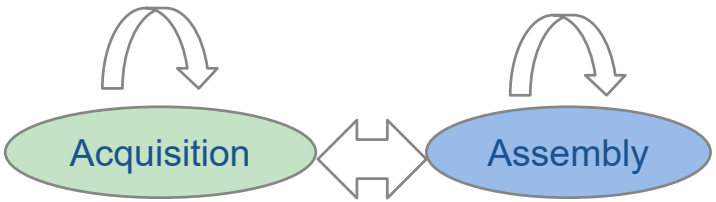


Schedule Summarization



- Probability of Task 2 being in a given state (or "meta task") depends on:
  - Whether Tasks 1 and 3 also 'belong' to the same meta task
  - The textual description of Task 2
  - Where in the original schedule the task occurs
- By treating this as a probabilistic problem, we can build a model capturing the analyst "intuition"
  - Enables building an algorithm for the same task—one that is highly accurate and efficient
- For simplicity, using only task name and prior task states (program stage) as inputs

Markov Chains

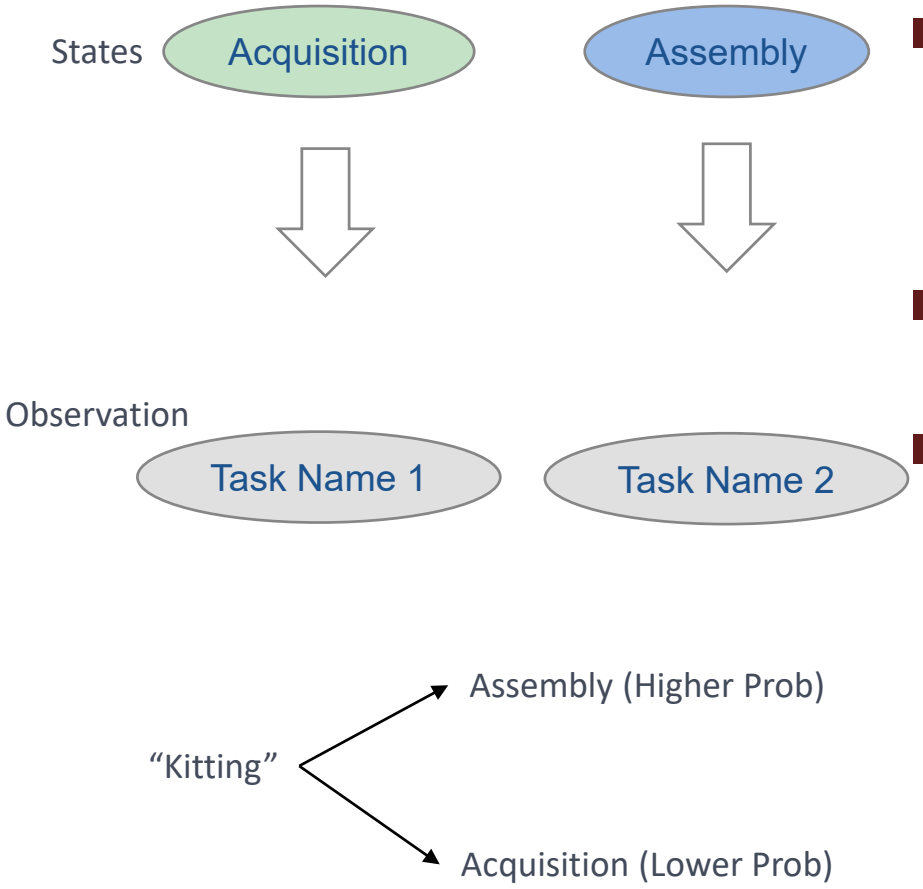


Transition Probability Matrix

	Acquisition	Assembly	Integration
Acquisition	0.1	0.4	0.5
Assembly	0.0	0.4	0.3
Integration	0.0	0.0	0.9

- A probabilistic model that characterizes the probability of transitioning between a set of observed states.
- Assumptions:
  - Conditional independence – probability of observation Z2 is dependent on Z1
  - The conditional independence assumption can be first order or many order (i.e., can be dependent on only the previous observation, or on the previous 3 observations)
- The parameters associated with a Markov Chain are the transition probabilities of the model
  - Results in a matrix of transition probabilities for some number of possible observations

Mixture Models



- Mixture distribution representing the probability distribution of observations in a population
- Analogous to clustering
  - K-means
- Parameters of the model are the probabilities of an observation given the underlying state
  - Results in a **matrix of emission probabilities**

Hidden States

What information is present in this DNA sequence?

ATGCTGCGATCGGTGGCCACCAGGGGGCGCCGACTGACTGACTGATCTAGCTACGTACGTACGTACGTAGCTACGAC

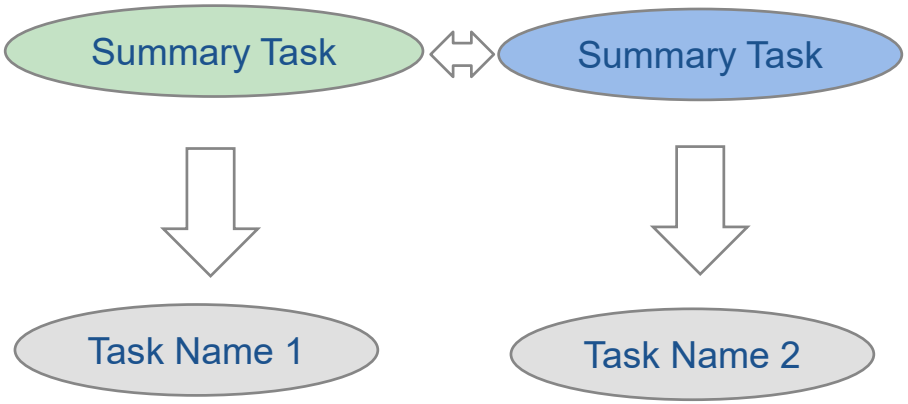
CTCF (Protein) Binding Site

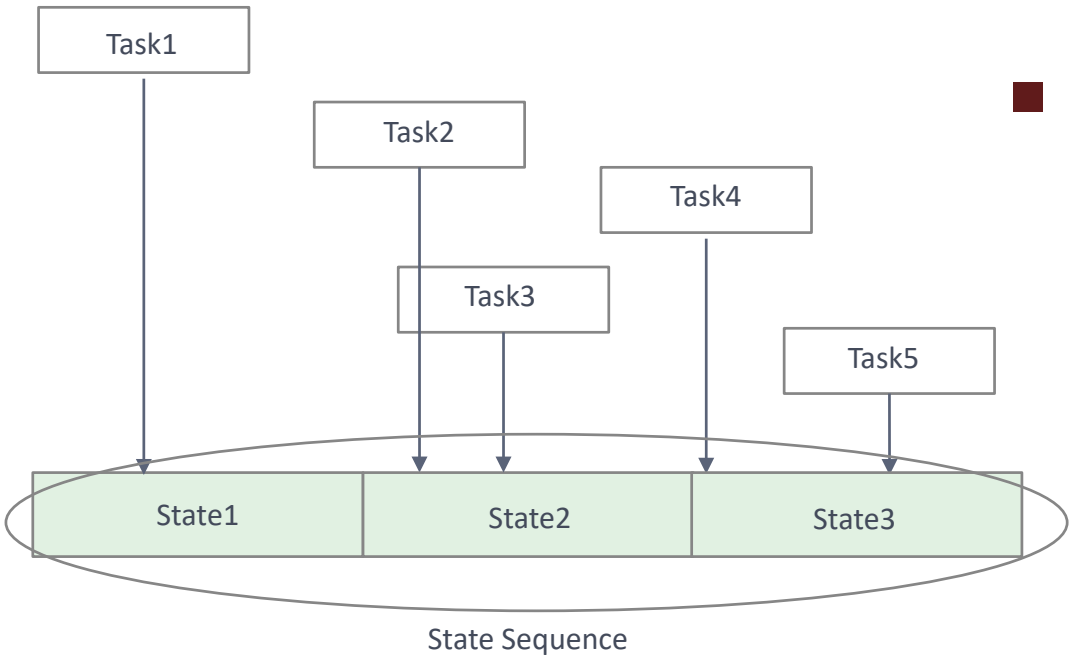
Hidden State

ATGCTGCGATCGGTGGCCACCAGGGGGCGCCGACTGACTGACTGATCTAGCTACGTACGTACGTACGTAGCTACGAC

Observation

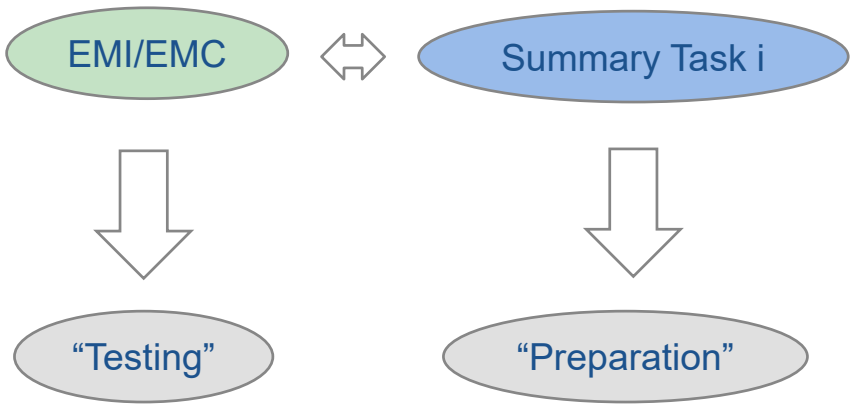
Hidden Markov Model (HMM)





- Models sequential or time-series data
- Parameterized by both state transitions (Markov Chain) and emission probabilities (Mixture Model)
- We use it to identify the state-sequence (labels/meta-tasks) associated with the observations

Decoding State Transitions



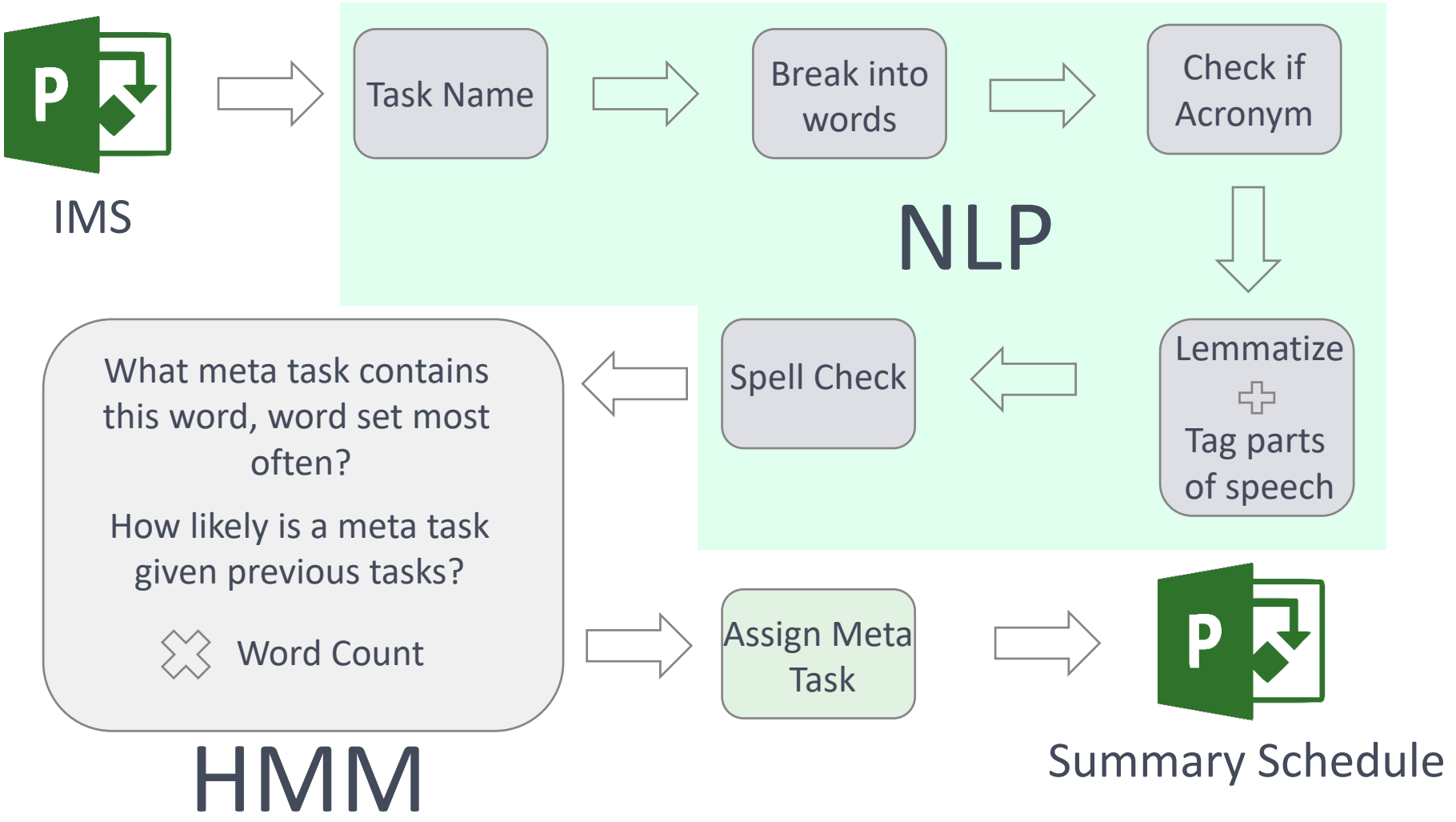
What is the probability Summary Task *i* is "Launch" given that the previous Summary Task was EMI/EMC and that we observe the word "preparation"?

EMI → Launch = 0.3

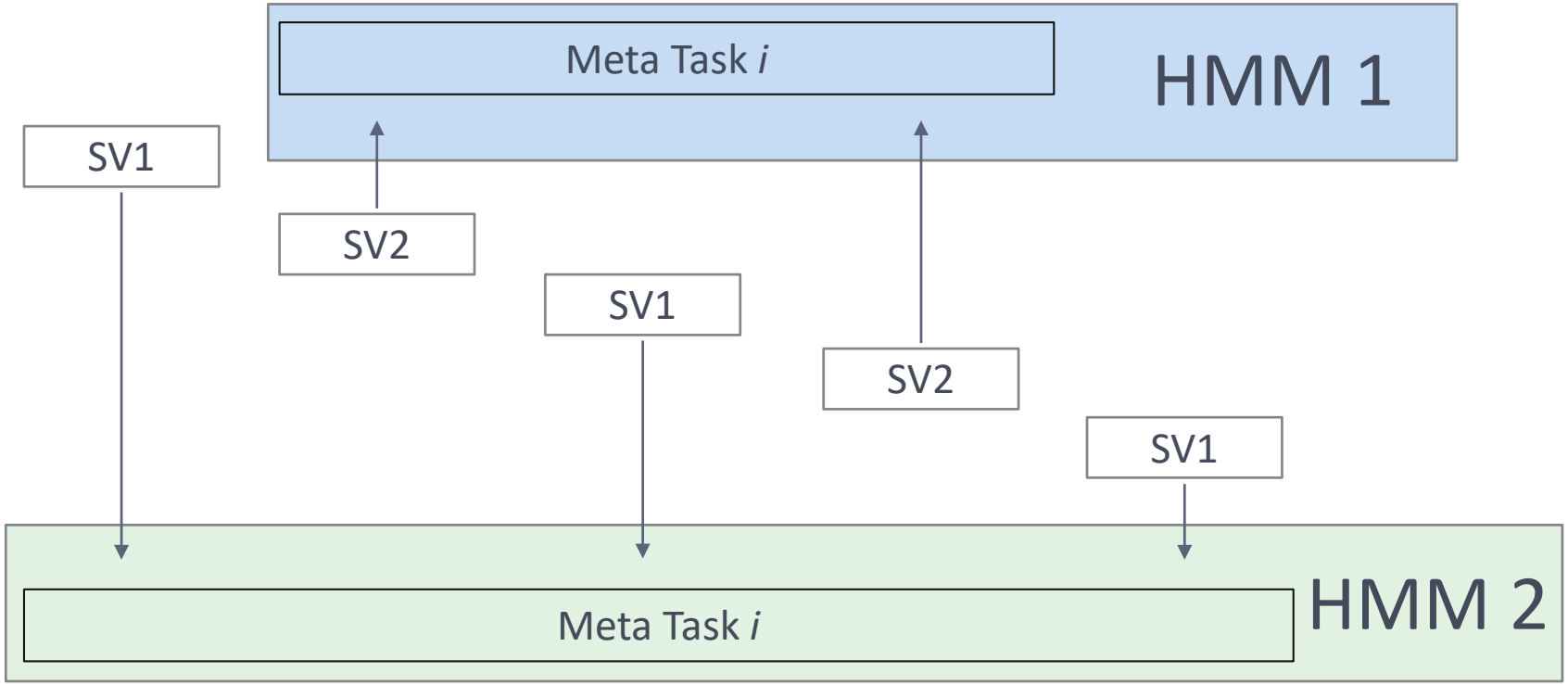
Preparation → Launch = 0.1

Launch = 0.1 × 0.3 = 0.03

Schedule Summarization Workflow



IMS Structure: Multiple Space Vehicles



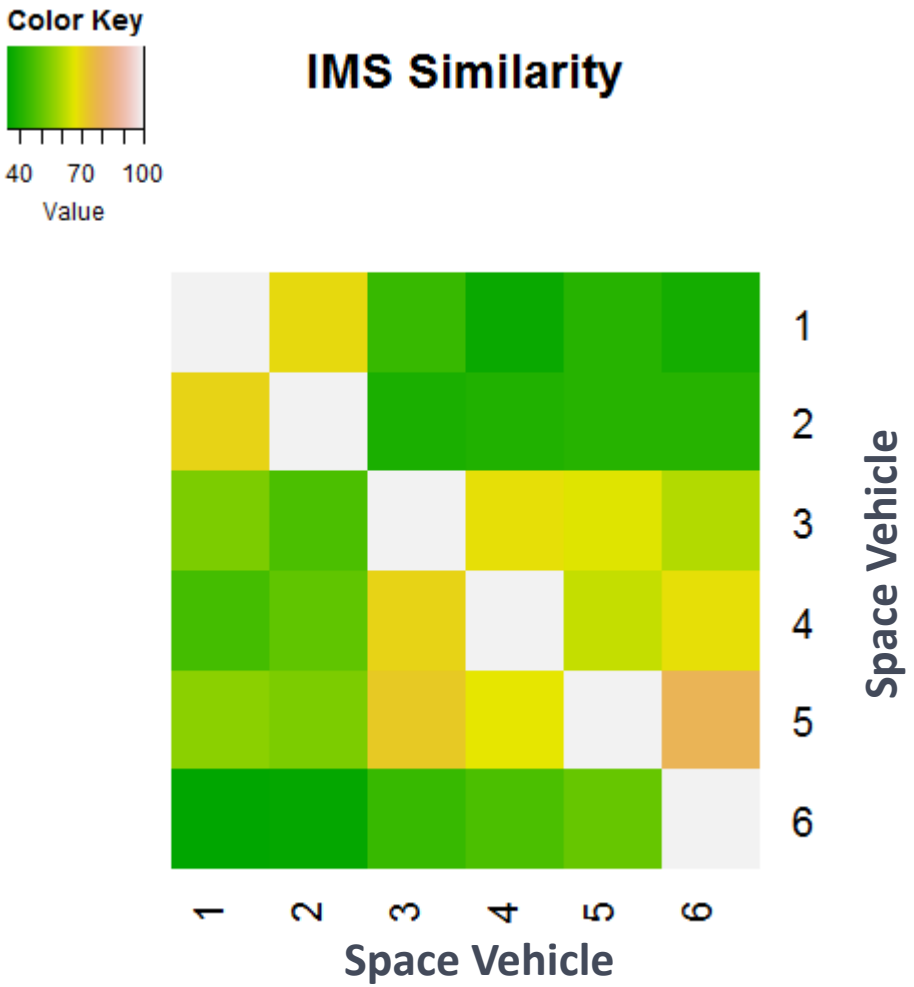
Interleaving of space vehicles, each with their own (partially) independent schedule and timescales means that each should be a part of a separate model.

Algorithm Testing Process

The following example uses an anonymous program with separate training and test datasets.

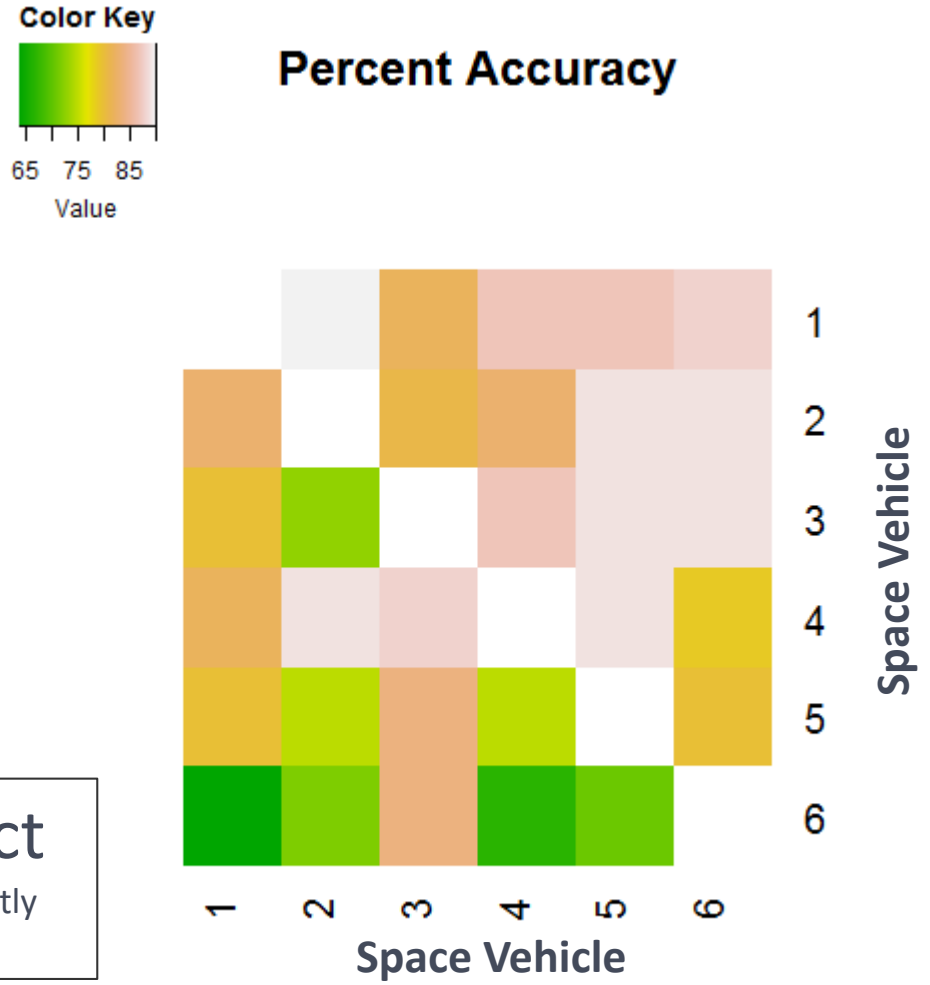
- Break an IMS out by Space Vehicle (SV)
  - Each sets of tasks associated with a SV are treated as a separate IMS
  - The tasks and the names for each SV differ, so it's a more realistic testing scenario
- Label the IMS tasks according to where they fall relative to key milestones
  - Utilizing milestone summary schedules
  - 7 labels (Acquisition, Assembly, Integration, Acoustic, TVAC, EMI/EMC, Launch)
- Training/testing scenarios:
  - Train on one SV, test on remaining SVs
  - Train on multiple SVs, test on 1
  - Use thresholds on minimum probability difference to transition between states

Similarity in Task Naming



Similarity is calculated by identifying the closest task in a comparison dataset and identifying how many have 1 or fewer differences.

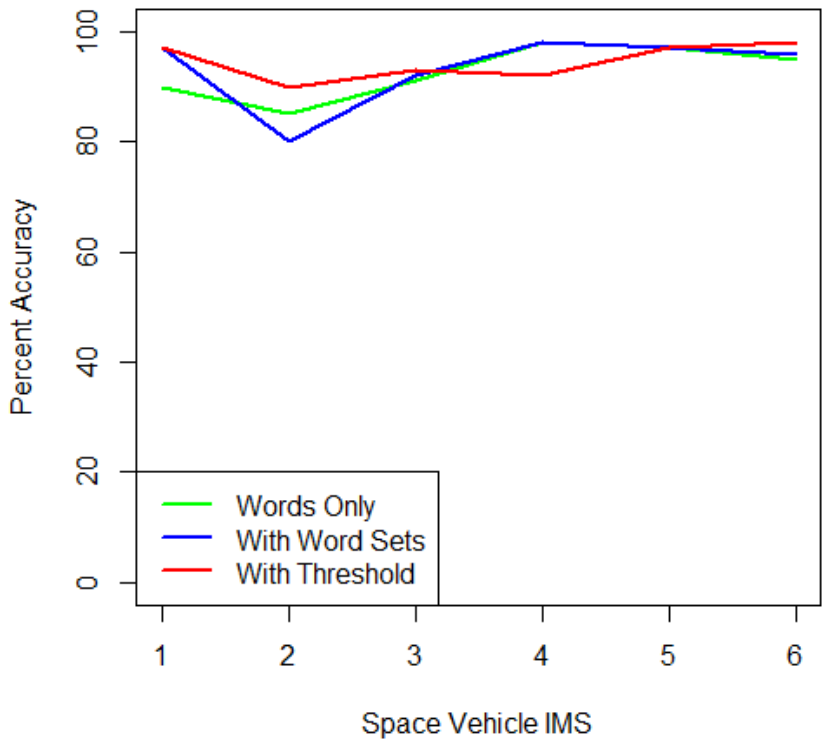
Testing Results: Training on One Space Vehicle



Percent Correct

(How many tasks are correctly assigned across an IMS)

Testing Results: Training on Multiple Space Vehicles



- Three tests:
  - Using individual words
  - Using word sets (two word combinations) and individual words
  - Using individual words, word sets, and thresholds
- Key Takeaways:
  - Accuracy as high as 98%
  - For SV2, thresholds improve accuracy to 90%
  - When using thresholds, accuracy is greater than 90% for all SVs

Summary

- Using Bayesian Networks (HMMs) and natural language processing, one can model the textual and sequential information embedded in an IMS.
- The model(s) generated allow for highly accurate mapping of IMS tasks to a summary schedule with sufficient training data
  - > 90% accuracy with more training data, > 80% percent with less training data

References

- Koller, D., Friedman, N., & Bach, F. (2009). *Probabilistic graphical models: principles and techniques*. MIT press.
- Durbin, R., Eddy, S. R., Krogh, A., & Mitchison, G. (1998). *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press.
- Helske, S., & Helske, J. (2017). Mixture hidden Markov models for sequence data: The seqHMM package in R. *arXiv preprint arXiv:1704.00543*.
- Craven, M., & Bockhorst, J. (2005). Markov networks for detecting overlapping elements in sequence data. In *Advances in Neural Information Processing Systems* (pp. 193-200).
- Biesinger, J., Wang, Y., & Xie, X. (2013, April). Discovering and mapping chromatin states using a tree hidden Markov model. In *BMC bioinformatics* (Vol. 14, No. 5, p. S4). BioMed Central.
- Bird, Steven, Edward Loper and Ewan Klein (2009), *Natural Language Processing with Python*. O'Reilly Media Inc.

Model Notation

- N = the number of states in the model
- Individual states are denoted $S = \{S_1, \dots, S_N\}$
- M = the number of possible observations, where observations are the task names
- Individual observations are denoted as $V = \{V_1, \dots, V_M\}$
- Transition probabilities (between states):
  - $A = \{a_{ij}\}$ where $1 \leq i, j \leq N$
  - $a_{ij} = P(q_{t+1} = S_i | q_t = S_j)$
  - $\pi = \{\pi_i\}$; $\pi = P(q_t = S_i)$ where $1 \leq i \leq N$
- Emission probabilities:
  - $E = \{e_{jk}\}$ where $1 \leq j \leq N$ and $1 \leq k \leq M$
  - $e_{jk} = P(v_k | q_t = S_j)$
- Log Likelihood (what we are maximizing!):
  - $\log L = \log P(V|\gamma)$ where $\gamma = \{\pi, A, E\}$
- Multi-model:
  - $\log L = \sum_{h=1}^{Z} \sum_{i=1}^{M} \log P(V_i|\gamma_h) P(\gamma_h)$

Why Summarize Schedules?

- You've been tasked to perform an independent Schedule Risk Assessment (SRA)
  - Goal: determine the projected completion date of a program
  - Summarization is used to provide an independent look at a program schedule
  - Reduce size 1,000s of tasks to 10-100s of tasks
- You've been tasked to perform an Joint Confidence Level (JCL)
  - Goal: determine the likelihood of achieving a specific completion date with a specific program cost
  - Summarization is used to:
    - Provide an independent look at the schedule
    - Provide a simplified structure to map cost data to

Summarizing schedules is difficult and time consuming!