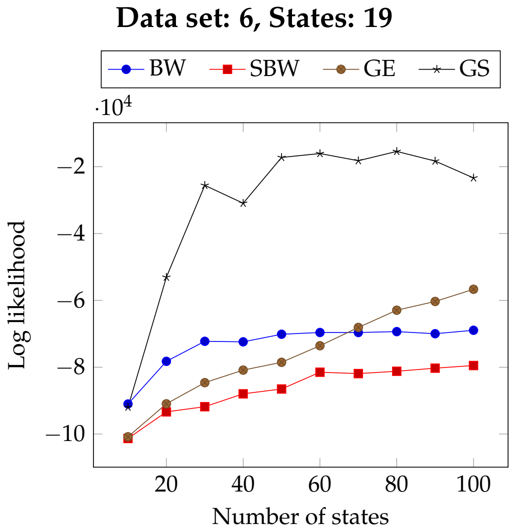
EXPERIMENTS

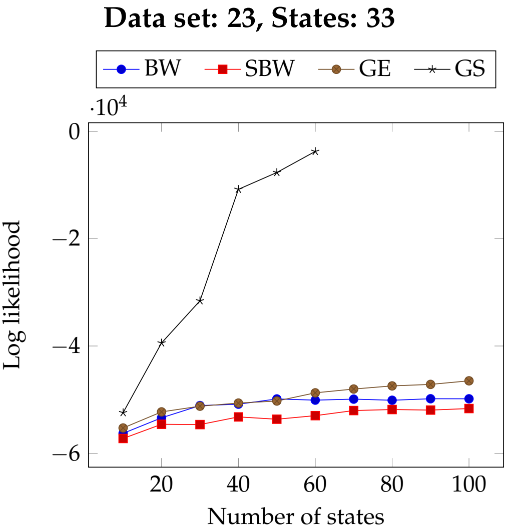
EXPERIMENTS

- Training - 5000 sequences
- Validation - 5000 sequences
- States - 10 to 100
- How does problem characteristics affect prediction accuracy
- Running time comparison

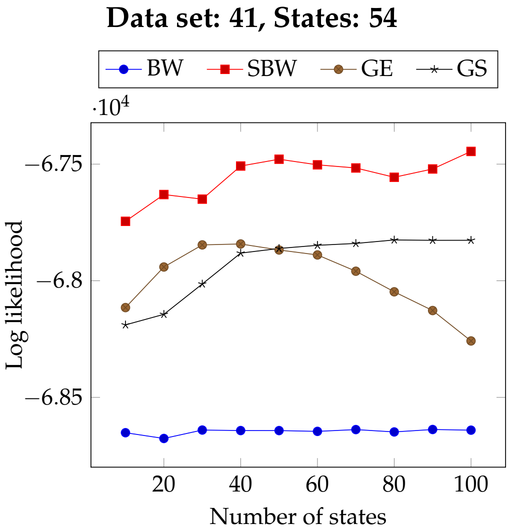
STATE SPACE


Data set: 6, States: 19
BW
SBW
GE
GS
·10^4
−2
−4
−6
−8
−10
20
40
60
80
100
Number of states
Log likelihood

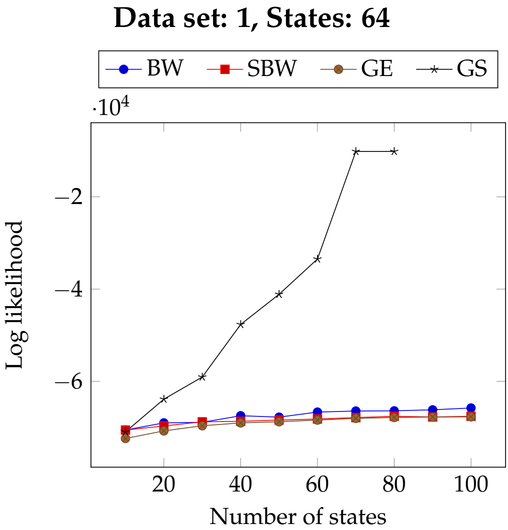
STATE SPACE


Data set: 23, States: 33
BW
SBW
GE
GS
$\cdot 10^4$
0
−2
−4
−6
20
40
60
80
100
Log likelihood
Number of states

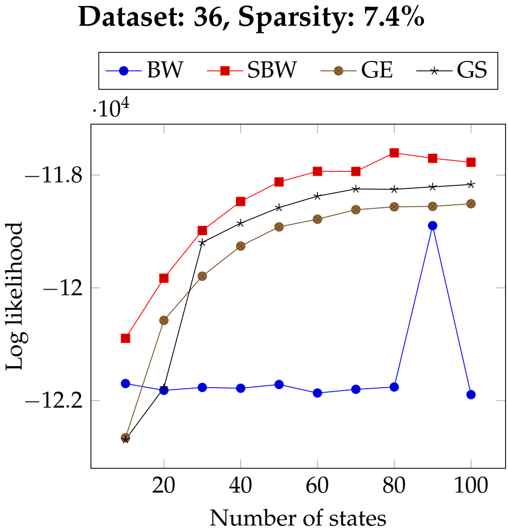
STATE SPACE


Data set: 41, States: 54
BW
SBW
GE
GS
$\cdot 10^4$
−6.75
−6.8
−6.85
20
40
60
80
100
Log likelihood
Number of states

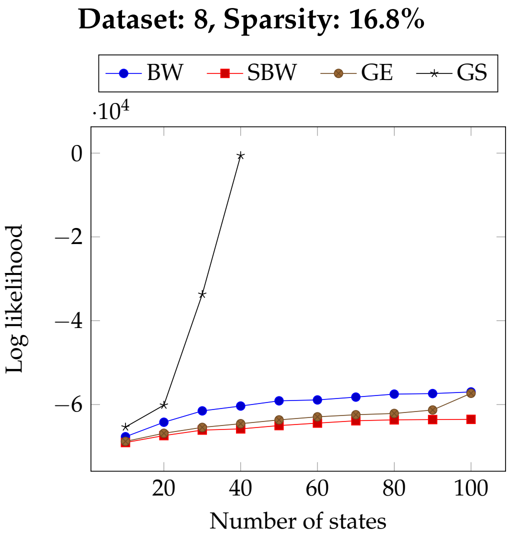
STATE SPACE


Data set: 1, States: 64
BW
SBW
GE
GS
·10^4
−2
−4
−6
20
40
60
80
100
Log likelihood
Number of states

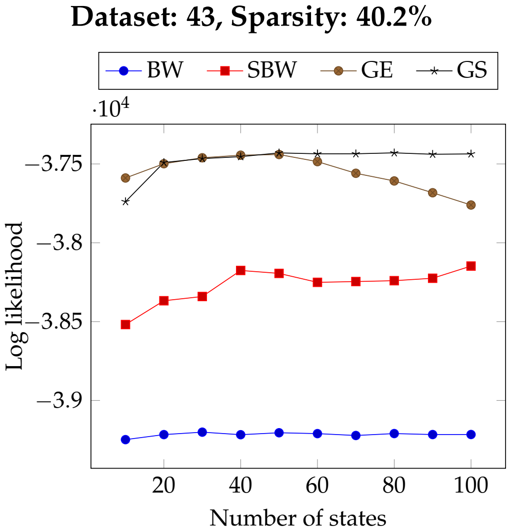
Transition sparsity



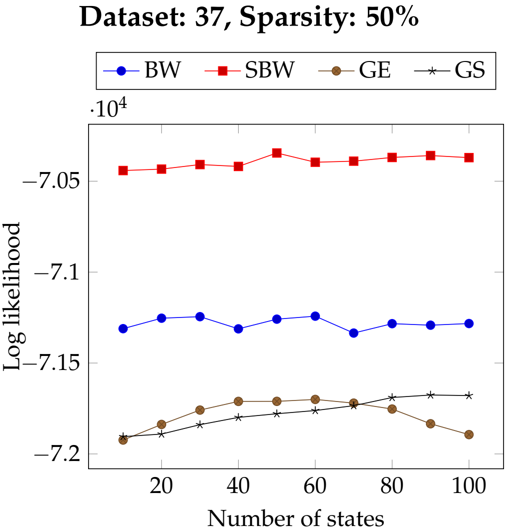
TRANSITION SPARSITY



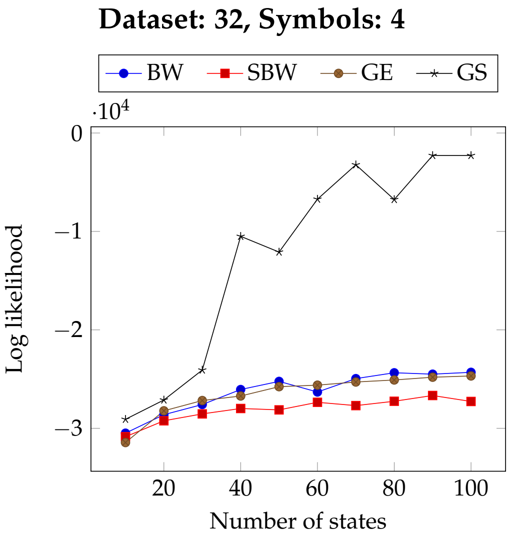
TRANSITION SPARSITY



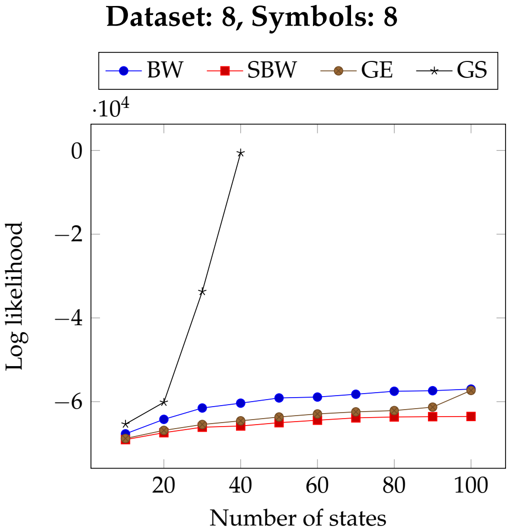
TRANSITION SPARSITY



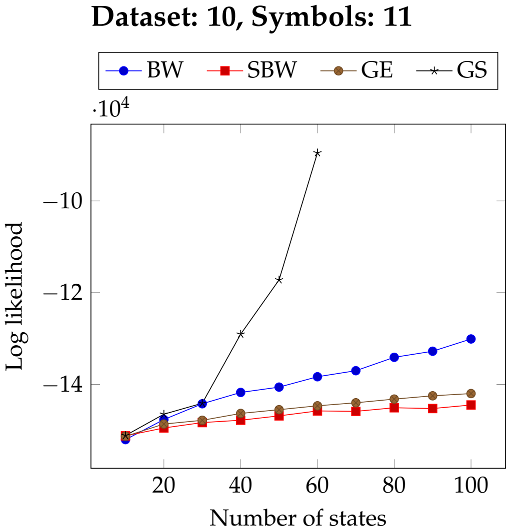
ALPHABET SIZE



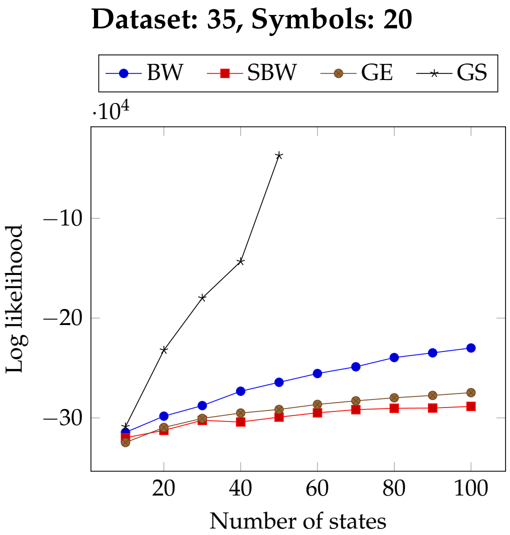
ALPHABET SIZE



ALPHABET SIZE



ALPHABET SIZE



SPEED COMPARISON

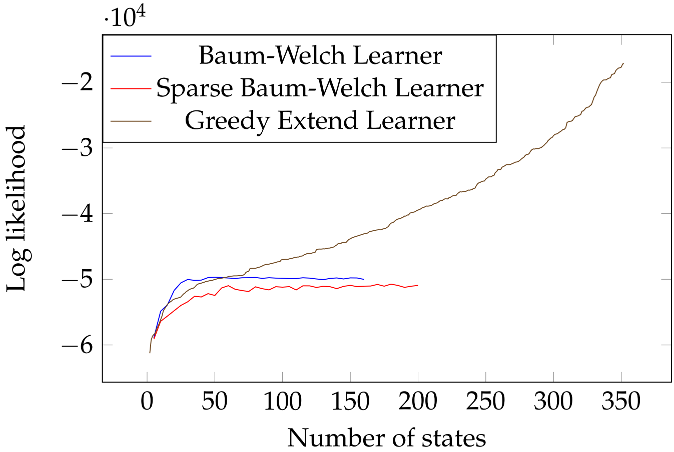


Figure: Results achieved in a time scope of eight hours.

SPEED COMPARISON

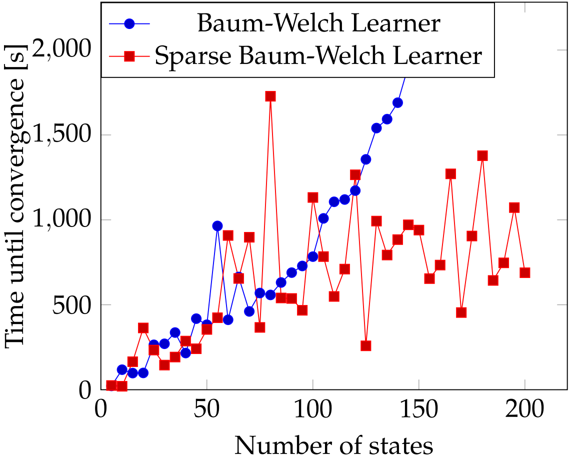


Figure: Running time comparison between Baum-Welch and Sparse Baum-Welch Learners

ALGORITHMS

STATIC VS. DYNAMIC

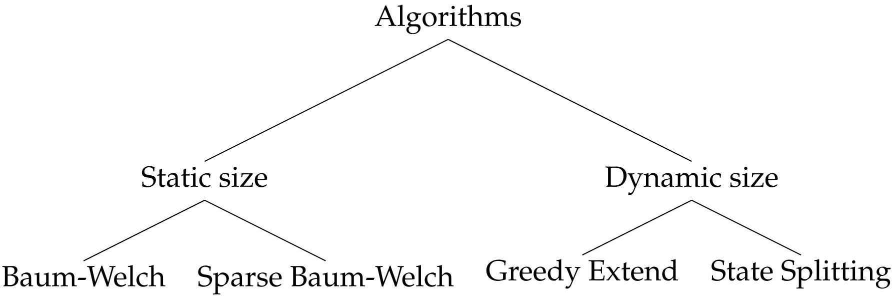


Figure: The algorithms used in our experiments

SPARSE BAUM-WELCH

- Creates HMM with n states and m symbols
- All parameters are initialized randomly
- Constraint: Each state has exactly log(n) transitions
- Other transitions set to zero
- Trained using Baum-Welch until convergence

Greedy Extend: Setup

- Works by adding states to the graph in iterations
- Starts as a single node with initial probability 1, random emission probabilities and a single transition to itself.

GREEDY EXTEND: ITERATIONS

1. Repeat until convergence
2. $G' = (V(G) \cup \{y'\}, E(G))$
3. Randomly choose a set X of $log|V(G')|$ nodes from $V(G')$
4. $\forall x \in X$ add transitions (x, y') and (y', x) to $E(G')$ with random probabilities
5. Normalize G'
6. if $LL(BW^{\beta}(G')) > LL(G)$, let $G = BW^{\beta}(G')$

STATE SPLITTING: OVERALL APPROACH

1. Identify a set of states $\mathcal{W}$ to split
2. Split all states in $\mathcal{W}$ using mechanic
3. Run Baum-Welch for β iterations

STATE SPLITTING: SPLITTING MECHANICS

- Clone Split
  - Makes a copy of the chosen state
  - Problem: BW unable to distinguish between clone and original
  - Alternative: Randomize clones probabilities.
- Distribution Split
  - Only splits if Transition or Emission probabilities are uniform.
  - Copies the emission probabilities from the original to the new state
  - Randomizes transition probabilities on the new state
  - Problem: Algorithm can get stuck (splits after 10 iterations)

STATE SPLITTING: IDENTIFICATION HEURISTICS

- The Heuristics compute a score ς that is used to choose which states to split
- Gamma Heuristic
  - Assign ς based on the number of times the state is visited when generating the sequence
  - $\forall i \in \{1, ..., n\} : \varsigma S_i = \sum_{O \in D} \sum_{t=1}^{T} \gamma_t(S_i)$

STATE SPLITTING: IDENTIFICATION HEURISTICS

- Viterbi Heuristic
    1. Compute $Q = \mathcal{V}_G(O)$ foreach signal $O \in D$
    2. Foreach state $s \in S$ determine its significant positions in Q
    3. $\forall s \in S \forall O \in D$ compute $\hat{\varsigma}_O(s) = \frac{\sum_{t \in \tau^O_{s,\lambda}} b_s(o_t)}{|\tau^O_{s,\lambda}|}$
    4. Compute $\forall s \in S \varsigma(s) = \sum_{O \in D} P(Q|O, \lambda) \hat{\varsigma}^O_{s,\lambda}$

EDGE CUTTING & STATE REMOVAL

- Edge cutting
  - Strict Edge cutting
  - Threshold Edge cutting
- State Removal

CHOSEN ALGORITHM

The algorithm we chose for the experiments had the following characteristics

- Splitting Mechanic: Distribution Split
- Identification: Gamma Heuristic
- β value: 10