1. Contribution
The target of this paper is to address a real-world problem of feature evolvable streams.

The contributions of this paper include:
1. A novel learning paradigm, named Feature Evolvable Streaming Learning (FESL), to model our problem;
2. A novel application of online learning techniques to solve FESL;
3. Empirical evaluations on both synthetic and real data sets.

2. Motivation
Common algorithms learned from streaming data assume an unchanged feature space. Unfortunately, this assumption does not hold in many streaming tasks.

For \( t = 1 \), \( \ldots \), \( T \), the learner observes \( x^i_t \in \mathbb{R}^d \) from feature space \( S_1 \); for \( t = T_1 + 1 \), \( \ldots \), \( T_2 \), the learner observes \( x^i_t \in \mathbb{R}^d \) from feature space \( S_2 \), respectively.

The baseline is to apply online gradient descent on every feature space.

\[
\text{loss}_i(t) = \mathbb{E}_\omega \left[ \nabla \omega_i \cdot (\omega_i - \hat{\omega}_i) \right], \quad i = 1, 2, \ldots
\]

However, the baseline suffers from two main deficiencies:
1. First, when new features just emerge, there are few data samples and thus, the training samples might be insufficient to train a strong model.
2. Second, the old model of vanished features is ignored, which is a big waste of our data collection effort.

3. Preliminaries
We only need to focus on one cycle and it is easy to extend to the case with multiple cycles.

- For \( t = 1 \), \( \ldots \), \( T_1 - T_2 \), the learner observes \( x^i_t \in \mathbb{R}^d \) from feature space \( S_1 \);
- For \( t = T_1 - 1 \), \( \ldots \), \( T_1 \), the learner observes both \( x^i_t \in \mathbb{R}^d \) from feature space \( S_1 \) and \( S_2 \), respectively.
- For \( t = T_2 + 1 \), \( \ldots \), \( T \), the learner observes \( x^i_t \in \mathbb{R}^d \) sampled from \( S_2 \).

4. The FESL Approach
- The two theorems imply that our two methods are comparable to the best predictor/classifier all the time.

Theorem 1. Assume that the loss function \( l \) is convex in its first argument and that it takes values in \([0, \ldots, 1]\). For all \( t \), \( T_1 \geq 1 \) and for all \( y \in \mathbb{Y} \) with \( t = T_1 - 1 \), \( T_1 + 1 \), \( \mathbb{L}^d \cdot \) with parameter \( \gamma = \sqrt{\frac{2}{d}} \), \( \alpha \) satisfies

\[
\mathbb{L}^d \cdot \leq \min \left\{ \mathbb{L}^d \cdot + \frac{k}{(T_1/2)^2} \right\}
\]

5. Experiments
- We present the trend of average cumulative loss. At each time \( t \), the loss \( \text{loss}_i(t) \) of each method is the average of the cumulative loss over \( 1, \ldots, t \), namely \( \text{loss}_i(t) = (1/t) \sum_{i=1}^{t} \text{loss}_i(t) \). The smaller the average cumulative loss, the better.

For synthetic datasets, FESL-c outperforms other methods on 8 datasets, FESL-s gets the best on 5. Our methods can follow the best baseline method or even outperform it.

For Reuters datasets, FESL-c outperforms other methods on 17 datasets, FESL-s gets the best on 9. Our two methods can take the advantage of NOGD and ROGD-1 and performance better than them.

6. Conclusion

For synthetic datasets, FESL-c outperforms other methods on 8 datasets, FESL-s gets the best on 5. Our methods can follow the best baseline method or even outperform it.

For Reuters datasets, FESL-c outperforms other methods on 17 datasets, FESL-s gets the best on 9. Our two methods can take the advantage of NOGD and ROGD-1 and perform better than them.