Imagine two identical people receive exactly the same training on how to classify certain objects. Perhaps surprisingly, we show that one can then manipulate them into classifying some test items in opposite ways, simply depending on what other test data they are asked to classify (without label feedback). We call this the Test-Item Effect, which can be induced by the order or the distribution of test items. We formulate the Test-Item Effect as online semi-supervised learning, and extend three standard human category learning models to explain it.

**The Test-Item Effects in Human Category Learning**

A computer can hold a trained classifier fixed during testing. A human cannot.

**Test-Item Effect**: Unlabeled test items change the classifier in human's mind. Two otherwise identical people A, B receiving exactly the same training data can be made to disagree on certain test items \(x\) simply by manipulating what other test data they are asked to classify, without label feedback.

**Test-Item Effect 1: Order of test items**

40 subjects, 1D feature space, 10 labeled items \(\{x=-2, y=0, (2,1)\}\) *5 Two conditions, 20 subjects each: 
- **L to R**: test item \(-2, -1.95, -1.9, \ldots, 2\)
- **R to L**: reverse order.

Results: Subjects in the “L to R” condition tend to classify more test items as \(y = 0\), and vice versa. For test items in \([-1.2, 0.1]\), a majority-vote among subjects will classify them in opposite ways.

**Test-Item Effect 2: Distribution of test items**

22 subjects, same feature space, 20 labeled items \((-1,0), (1,1)\) *10 Test items drawn from two-component GMM. Two conditions:
- **L shifted**: GMM means at \(-1.43\) and 0.57
- **R shifted**: GMM means at -0.57 and 1.43

Results: early (in first 50 test items) decision boundaries shifted according to condition and late (after 700 test items) boundaries shifted according to condition.

**Test-Item Effects as Online Semi-Supervised Learning**

The key is to update classifier upon unlabeled data. Standard human category learning models in psychology (equivalent to supervised learning models) cannot explain test-item effects. We propose and compare three online semi-supervised extensions:

- **Semi-Supervised Exemplar Model**
  - Self-training Nadaraya-Watson kernel estimator; extends the generalized context model (Nosofsky, 1986)
  - Parameter: kernel bandwidth \(h\)
  - Algorithm 1: Semi-Supervised Exemplar Model
    1. For \(n = 1, 2, \ldots\)
    2. Receive \(x_n\), predict its label by thresholding \(\tau(x_n) = \frac{\sum_{i=1}^{m} k(x_n, x_i)}{\sum_{i=1}^{m} k(x_n, y_i)} \geq 0.5\)
    3. Receive \(y_n\), (may be unlabeled), update model:
       - If \(y_n\) is unlabeled then
         - \(\hat{y}_n = \tau(x_n)\)
       - else
         - \(\hat{y}_n = y_n\)
    4. end for

- **Semi-Supervised Prototype Model**
  - Incremental EM on GMM (Neal & Hinton, 1998), but without revisiting old items; extends (Posner & Keele, 1968)
  - Parameters: Prior of \(\theta\)
  - Algorithm 2: Semi-Supervised Prototype Model
    1. For \(n = 1, 2, \ldots\)
    2. Receive \(x_n\), classify by \(y = \tau(x_n, \theta^{n-1})\)
    3. Receive \(y_n\), (may be unlabeled), update model:
       - E-step:
         - If \(y_n\) is unlabeled then
           - \(\phi = \phi + R(\tau(x_n, y))\)
         - else
           - \(\phi = \phi + R(x_n, y_n)\)
    4. M-step:
       - Let \(\theta = (\mu_1, \mu_2, \sigma_1, \sigma_2)\), Compute \(\theta^{n+1}\) as follows:
         - \(\alpha = \frac{\phi(x_n, y, \theta)}{\phi(x_n, y, \theta^{n-1})}\)
         - \(\mu_1 = \phi(x_n, y, \theta) \cdot \mu_1 + \alpha \phi(x_n, y, \theta^{n-1}) \cdot \mu_2\)
         - \(\sigma_1 = \phi(x_n, y, \theta) \cdot \sigma_1 + \alpha \phi(x_n, y, \theta^{n-1}) \cdot \sigma_2\)
       - end for

- **Semi-Supervised Rational Model of Categorization**
  - Dirichlet Process Mixture Model with marginalization over \(y\); extends (Anderson, 1990)
  - Algorithm 3: Semi-Supervised Rational Model of Categorization
    1. For \(n = 1, 2, \ldots\)
    2. Receive \(x_n\), (may be unlabeled)
    3. Re-sample \(m\) particles
    4. Predict \(y_n\) with new particles
    5. end for