Discriminately Decreasing Discriminability with Learned Image Filters

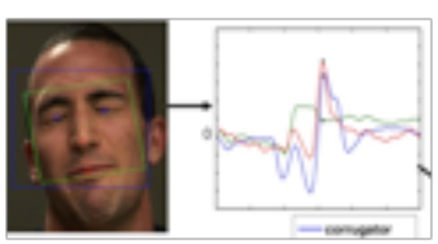
Jacob Whitehill[1,2] and Javier Movellan[2]

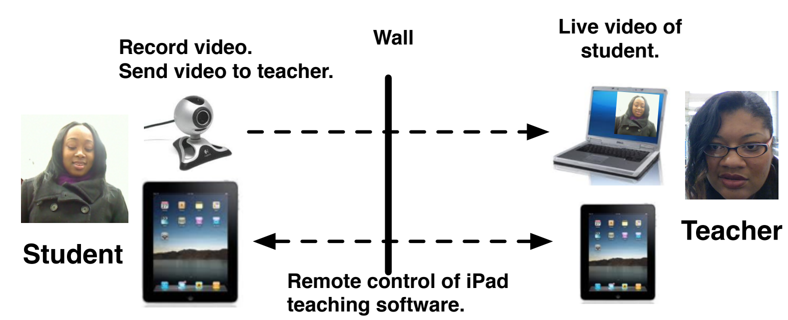
[1]Computer Science & Engineering
[2]Machine Perception Laboratory
UCSD

UCSD AI Seminar
26 Sep 2011

Machine Perception Lab

- Research activities:
  - Study *natural human behavior* from a computational perspective.
  - Develop *machine sensors* to mimic the perceptual power of humans.
  - Create *intelligent systems* that interact with humans, e.g., social robots, automated teaching systems.





Machine Perception Lab

- Most of our projects require lots of *labeled training data*:
  - Computer Expression Recognition Toolbox (CERT):
    - Tool for fully automatic real-time facial expression recognition from images/video.
    - Face detector: ~100,000 training images labeled for face location.
    - Expression classifiers: ~10,000 face images labeled for ~50 facial attributes each.

Machine Perception Lab

- Most of our projects require lots of *labeled training data*:
    - Automated teaching system of math/logic skills:
        - Automatic "mood" detectors: ~50,000 face images labeled 1-4 for engagement, confusion, frustration, etc.

Data labeling

- Traditionally, data were labeled by *hiring undergraduate students*. 
  - Expensive, slow: thousands of dollars over 4 months to collect 60,000 smile/non-smile labels.
- More recently, *crowdsourcing* services such as ESP Game, HerdIt, and Amazon Mechanical Turk have been used. amazonmechanical turk beta Artificial Artificial Intelligence
  - Cheap, fast: $200 over 1 week to collect 1,000,000 smile/non-smile labels.

Data labeling

- Unfortunately, crowdsourcing suffers from two problems:
  1. *Unreliable* -- the labelers' accuracy may be questionable.
     Welinder, et al., 2010; Ruvolo, et al., 2010; Whitehill, et al., 2009
  2. *Insecure* -- the data may be too sensitive to distribute widely, e.g.:
     - Identity of a face image, e.g.,
       students' faces in automated teaching study.
       - Some students in our experiments are portrayed in unflattering ways (e.g., crying).
     - Geographical location of a satellite image.

Filtering out identity

- What if we could *filter* the images/videos to "remove" the person's identity (*face de-identification*; Newton, et al. 2005), yet preserve the attribute to be labeled?
    - Crowdsourcing might then be viable, as the "sensitive information" is erased.
    - For simple applications, we could try constructing the filter manually...

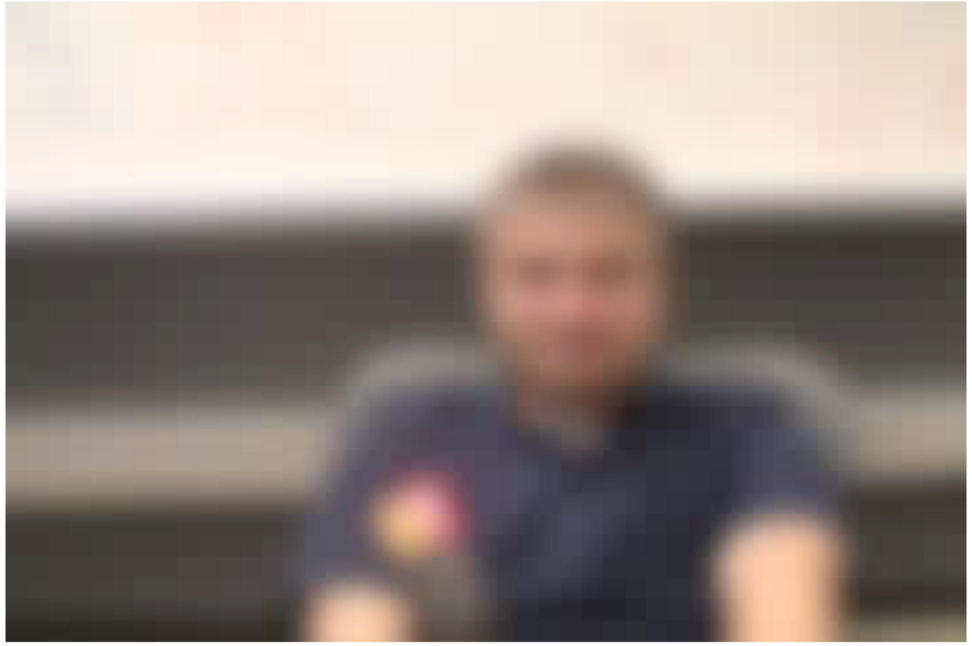
Filtering out identity

- We explored this idea on video data already collected for a study on *driver fatigue* (Vural, et al. 2008).
- In videos on the next slide, subjects are playing a *race car driving game*.
- Videos were filtered using hand-selected Gaussian blur filter (σ = 12 pixels).
- In which video does the subject appear *more fatigued* (2AFC task)?

Which shows more fatigue?

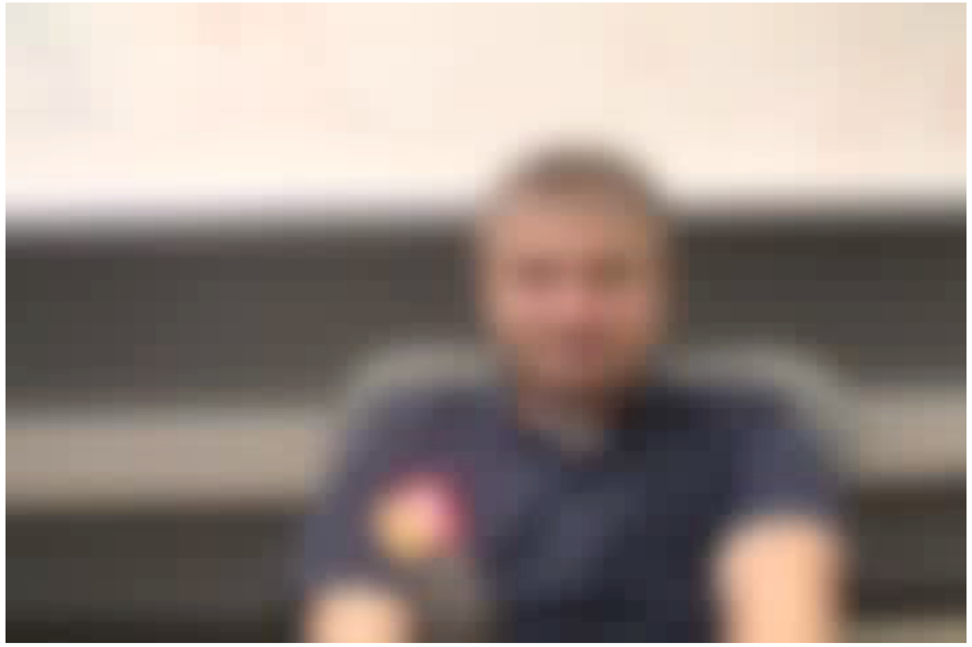
(a) (b)

Which shows more fatigue?



(a) (b)

Which shows more fatigue?

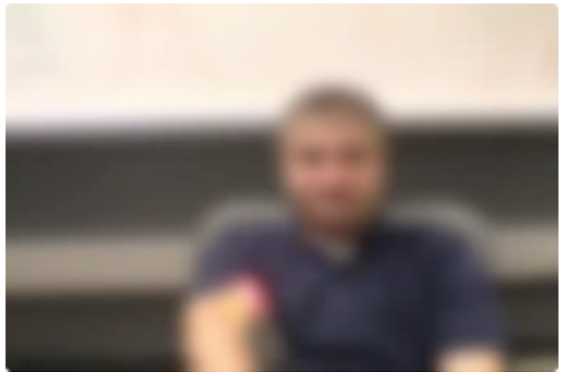


(a)

(b)

Filtering our identity

- In pilot experiment, labels from 69 MTurk labelers agreed 100% (after taking majority vote) on 55 *blurred* videos compared to *original* videos when labeling "more/less engaged".
  - I.e., fatigue is still discriminable despite blur.
- As intended, much of the identity information is suppressed by the filter.
  - Identity is less discriminable.



Filtering out identity

- This pilot study suggested that "filtering out identity" is possible.
- However, it was also too "easy":
  - Fatigue is contained in low-frequency components.
  - Identity is contained in high-frequency components.
  - A simple low-pass (Gaussian) filter works well.
- What about more general settings?
  - Is it possible to design the filter automatically?

Filtering our identity

- What we want is to discriminately decrease discriminability:
  - *Decrease* discriminability of identity.
  - *Preserve* discriminability of attribute-of-interest.
- Note that discriminability can apply both to *human perception* as well as *machine classification*.
  - In this work, we address both types of discriminability.

Discriminately
decreasing
discriminability

Discriminately decreasing discriminability

- We approached the task of discriminately decreasing discriminability (“DDD”) as an optimization problem.
  - *Input*: a set of data points, each of which is labeled for a “target” task A and a “distractor” task B.
  - *Output*: a filter θ that maximally *increases* discriminability of task A, while maximally *decreasing* discriminability of task B.

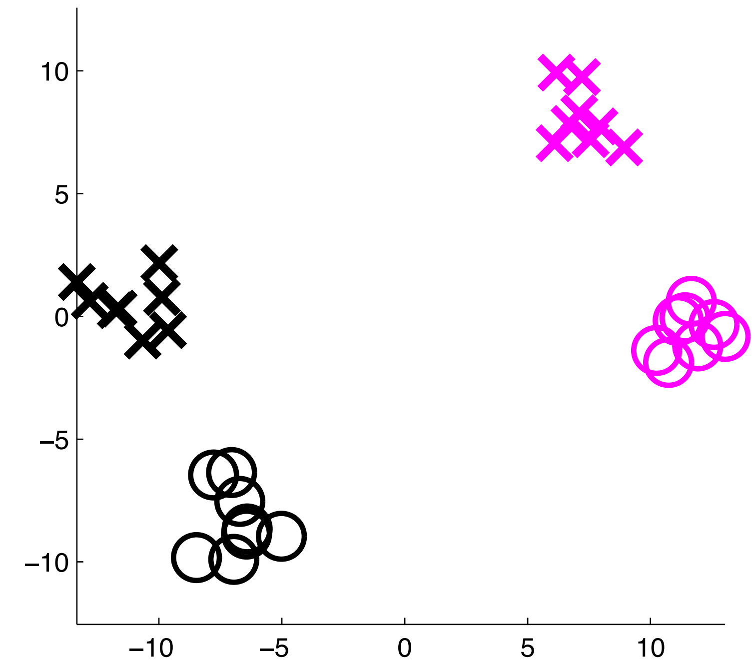
Discriminately decreasing discriminability

- We focus on the case of *binary* labeling tasks, e.g.:
  - Student appears engaged/not engaged.
  - Person is smiling/not smiling.
  - Person is male/female.
- Binary labels do not directly capture *identity*.
  - However, it turns out that suppressing *gender* seems to implicitly suppress identity as well (more later).

Simple example in R^2

- Consider data $\{ x_i \}$ in R^2:
  - Binary labeling Task A: magenta-versus-black
  - Binary labeling Task B: O-versus-X.
- Both labeling tasks A and B are both highly discriminable.

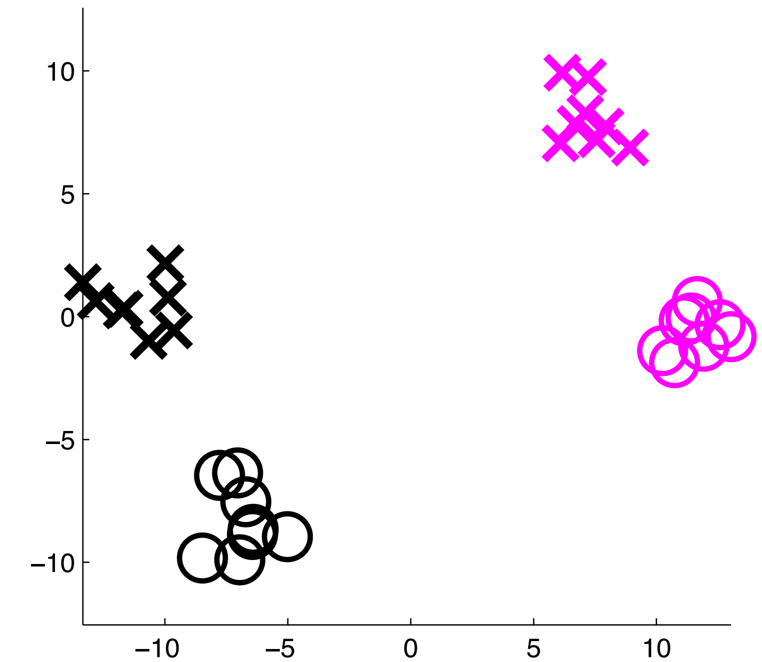
Original data



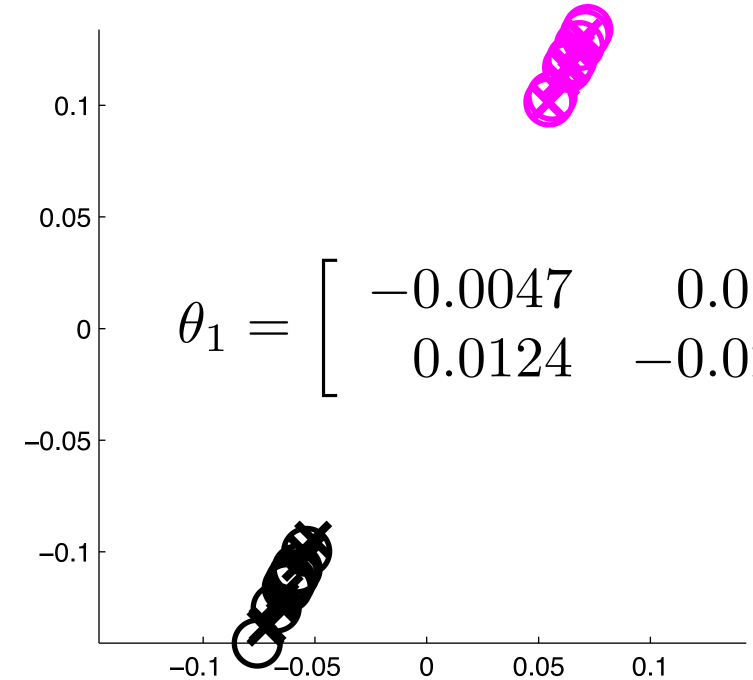
Simple example in R^2

- Suppose we wish to *preserve* discriminability of Task A (magenta-versus-black), but *suppress* discriminability of Task B (O-versus-X).
- We can filter the $\{ x_i \}$ with some filter θ:
    - In this case, $F(\theta, x) = \theta x$ where θ is a 2 x 2 matrix (general linear transformation).
    - Task A (black versus magenta) is still highly discriminable, but Task B is not.

Original data



Filtered data

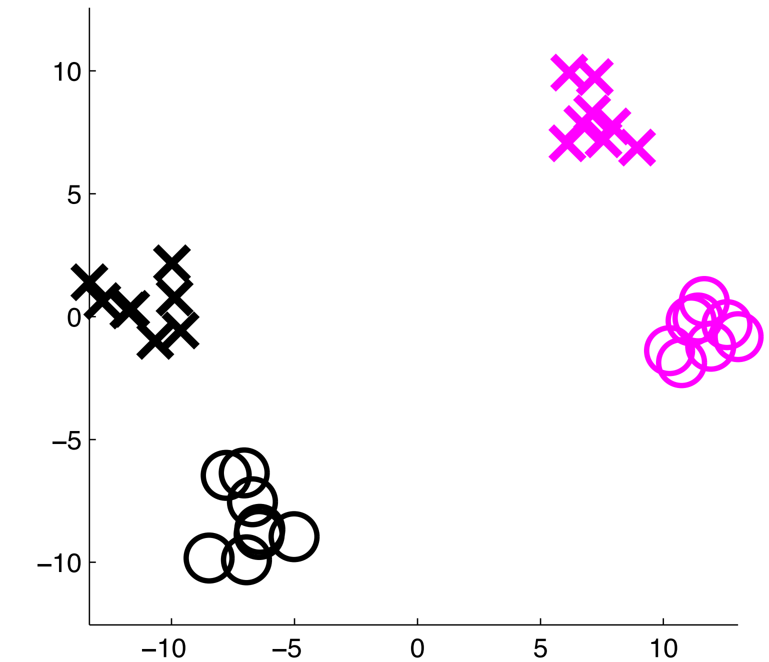


$$\theta_1 = \begin{bmatrix} -0.0047 & 0.0107 \\ 0.0124 & -0.0285 \end{bmatrix}$$

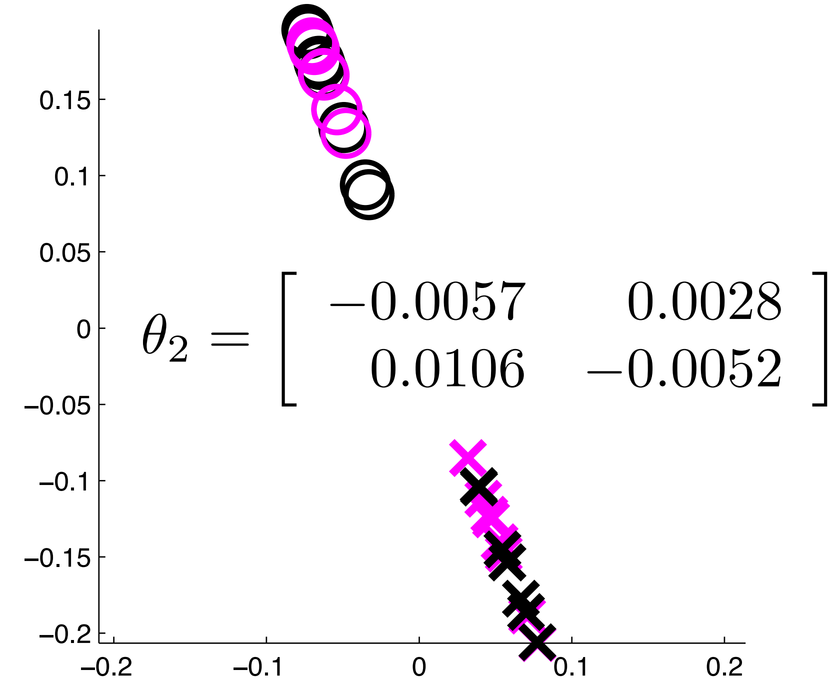
Simple example in R^2

- Alternatively, we can apply a filter that preserves discriminability of Task B (O-versus-X) while decreasing discriminability for Task A (magenta-versus-black).
- How can we learn such filters θ_1 and θ_2 automatically?

Original data



Filtered data



$$\theta_2 = \begin{bmatrix} -0.0057 & 0.0028 \\ 0.0106 & -0.0052 \end{bmatrix}$$

Discriminately decreasing discriminability: formalization

- Four inputs:

1. $\{ x_i \}$, where each column-vector $x_i \in R^d$
   - Each x_i might be an image with d pixels.
2. $L_a: R^d \rightarrow \{ 0, 1 \}$, "target" task
   $L_b: R^d \rightarrow \{ 0, 1 \}$ "distractor" task
   - $L_a(x)$ might represent whether a face image x is smiling/not smiling.

     $L_b(x)$ might represent whether a face image x is male/female.

Discriminately decreasing discriminability: formalization

- From { x_i } and L_a, L_b, we can define four matrices (each with d rows), each containing some of the data points:
  - X_{0a}: contains all x_i for which $L_a(x_i) = 0$
    X_{1a}: contains all x_i for which $L_a(x_i) = 1$
  - X_{0b}: contains all x_i for which $L_b(x_i) = 0$
    X_{1b}: contains all x_i for which $L_b(x_i) = 1$

Discriminately decreasing discriminability: formalization

- Four inputs (continued):

3. A filter function $F(\theta, X)$ that filters each data point x in matrix X.

4. Some "discriminability metric" $D(F(\theta, X_0), F(\theta, X_1))$ which measures the real-valued "discriminability" of filtered data in X_0 from filtered data in X_1.

- Then, our goal is to find θ for which:
  - $D(F(\theta, X_{0a}), F(\theta, X_{1a}))$ is *large*. "target" task
  - $D(F(\theta, X_{0b}), F(\theta, X_{1b}))$ is *small*. "distractor" task

Discriminately decreasing discriminability: formalization

- One way of finding such a θ is to optimize the *negative ratio of discriminabilities*, *R*(θ), of Tasks A and B:

$$R(\theta) = -\log \frac{D(F(\theta, X_{0a}), F(\theta, X_{1a}))}{D(F(\theta, X_{0b}), F(\theta, X_{1b}))} + \beta\theta^\top\theta$$

  where β is the regularization strength on θ.

  - *R* is small when discriminability of Task A is *large*, and when discriminability of Task B is *small*.

- We can then minimize *R* w.r.t. filter parameters θ.

$$\theta^* = \arg\min_\theta R(\theta)$$

Discriminately decreasing discriminability: formalization

- As long as *D* is differentiable in *F*, and *F* is differentiable in θ, then we can find a local minimum θ^* of *R* using gradient descent.
  - For a variety of filters, the function derivative of *F* w.r.t. θ can be found analytically.
    - E.g., convolution filters, general linear transformations, and pixel-wise "mask" filters are all linear in θ and X.
  - But how do we define the "discriminability metric" *D*?

Discriminability metric *D*

- One notion of discriminability is the *margin* of an SVM (shortest distance to separating hyperplane).
- Hence, to compute $D_{svm}(F(\theta, X_0), F(\theta, X_1))$, we could train an SVM on the filtered data points, and then compute the margin.

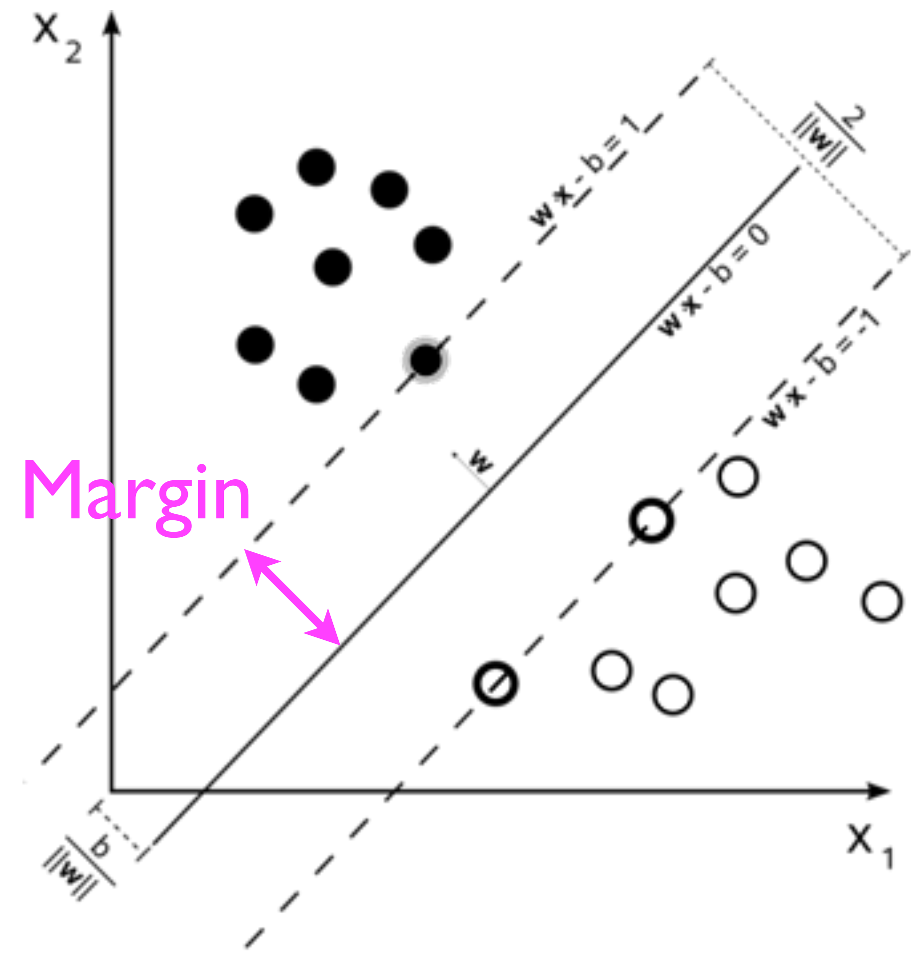


Image courtesy of Wikipedia.

Discriminability metric *D*

- One notion of discriminability is the *margin* of an SVM (shortest distance to separating hyperplane).
- Hence, to compute $D_{svm}(F(\theta, X_0), F(\theta, X_1))$, we could train an SVM on the filtered data points, and then compute the margin.
- Problem: the optimal hyperplane, and hence D_{svm}, must be found *numerically* by solving a quadratic programming problem.
    - Hence, the derivative of D_{svm} is not available in closed form.

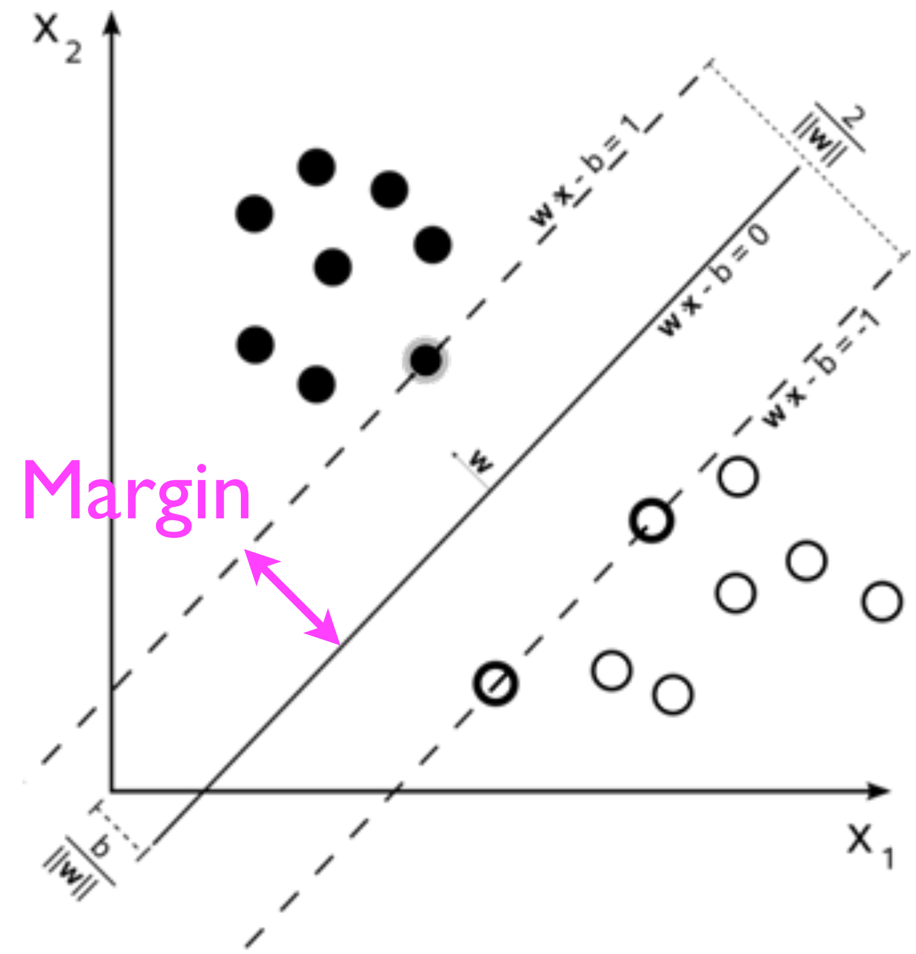


Image courtesy of Wikipedia.

Discriminability metric *D*

- We need a discriminability metric that can be found in *closed form* and that is differentiable in *F*.
- Here, the classic method Linear Discriminant Analysis (LDA) (Fisher 1936) is useful:
  - LDA projects each point onto a vector *p* and then computes the *ratio of between-class variance to within-class variance* (sometimes called *J*):

Between-class variance

$$J(p, X_0, X_1) = \frac{p^\top(\overline{x}_0 - \overline{x}_1)(\overline{x}_0 - \overline{x}_1)^\top p}{p^\top[(X_0 - \overline{X}_0)(X_0 - \overline{X}_0)^\top + (X_1 - \overline{X}_1)(X_1 - \overline{X}_1)^\top]p}$$

Within-class variance

$\overline{x}_0$ is mean vector of class 0.
$\overline{X}_0$ contains n_0 copies of $\overline{x}_0$, where n_0 is number of data labeled 0.

  - In LDA, the separating hyperplane is defined to have normal vector p^* that maximizes *J* for X_0 and X_1.

Discriminability metric *D*

- LDA is useful because the maximum of *J*, as well its argmax p^*, can both be found *analytically*:

$$
\begin{aligned}
J^*(X_0, X_1) &= \max_p J(p, X_0, X_1) \\
p^* &= \arg\max_p J(p, X_0, X_1) \\
&= [(X_0 - \overline{X}_0)(X_0 - \overline{X}_0)^\top + (X_1 - \overline{X}_1)(X_1 - \overline{X}_1)^\top]^{-1}(\overline{x}_0 - \overline{x}_1)
\end{aligned}
$$

- We then define our discriminability metric *D* in terms of the the "maximum Fisher discriminability" J^* of the *filtered* data:

$$D_{\mathrm{lda}}(F(\theta, X_0), F(\theta, X_1)) = J^*(F(\theta, X_0), F(\theta, X_1))$$

- Through straightforward linear algebra, we can find a closed-form expression for the derivative of D_{lda} w.r.t. *F*.

Discriminately decreasing discriminability: formalization

- Using D_{lda} as the discriminability metric, we can optimize the objective function *R* w.r.t. θ so that Task A is highly discriminable while Task B is not:

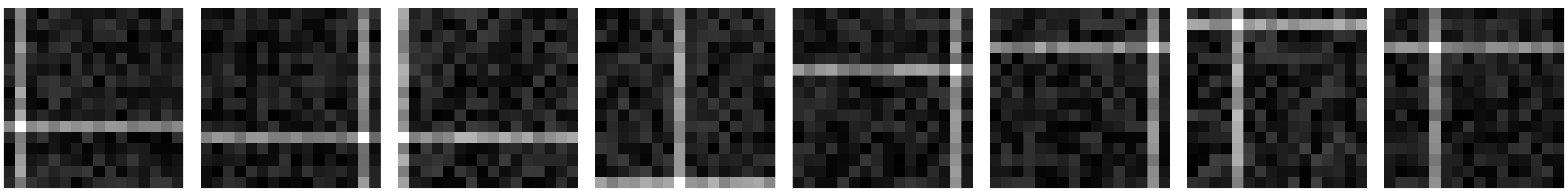
$$R(\theta) \quad = \quad \log \frac{D(F(\theta, X_{0a}), F(\theta, X_{1a}))}{D(F(\theta, X_{0b}), F(\theta, X_{1b}))} + \beta\theta^\top\theta$$

We abbreviate this gradient ascent procedure as "DDD" (discriminately decreasing discriminability).

Experiments

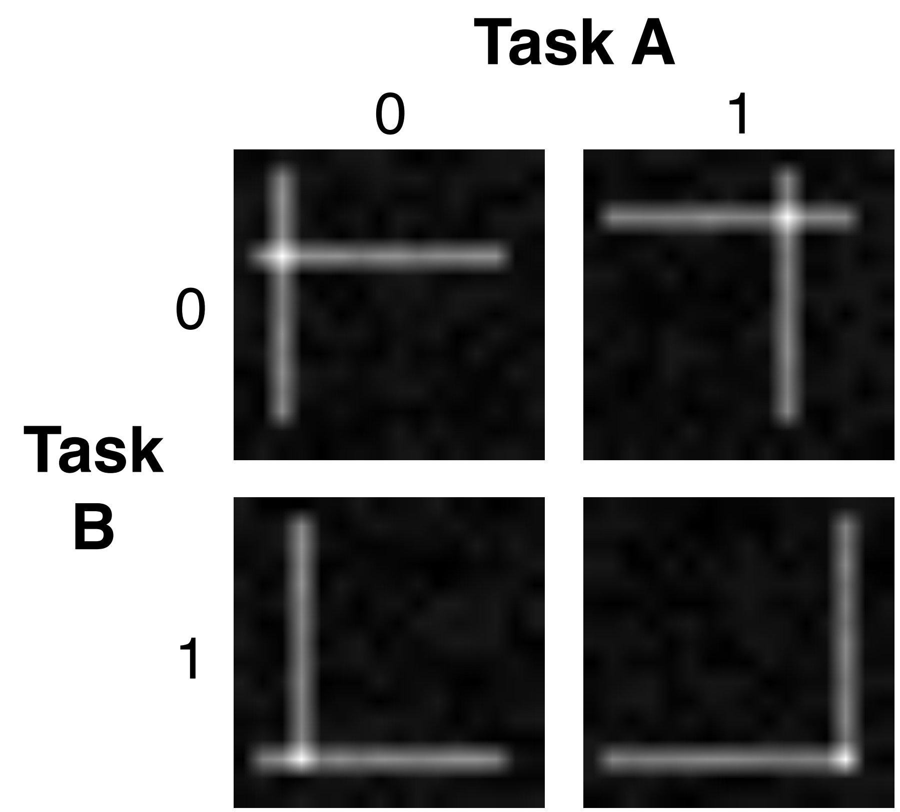
Experiment 1: Crossing lines

- As a proof-of-concept experiment, we generated 1000 images (16x16 pixels) consisting of 1 vertical line + 1 horizontal line + uniform noise:



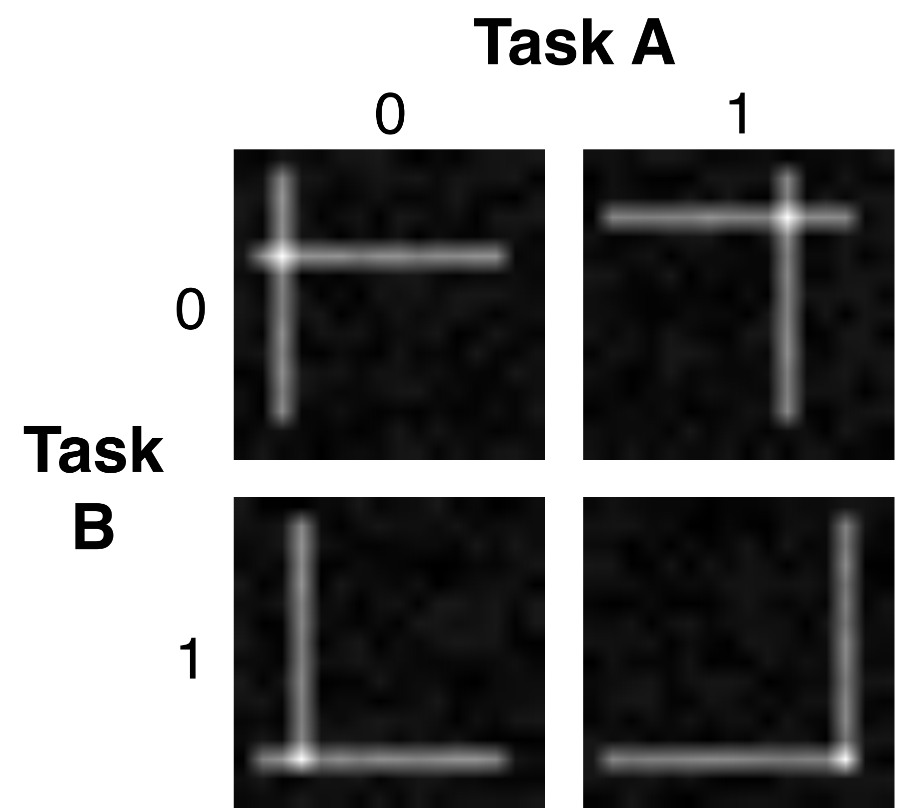
Experiment 1: Crossing lines

- We then defined Task A and Task B as follows:
    - Task A:
        - *x* is class 0 if vert. line is in *left half* of image.
        - *x* is class 1 if vert. line is in *right half* of image.
    - Task B:
        - *x* is class 0 if horz. line is in *top half* of image.
        - *x* is class 1 if horz. line is in *bottom half* of image.



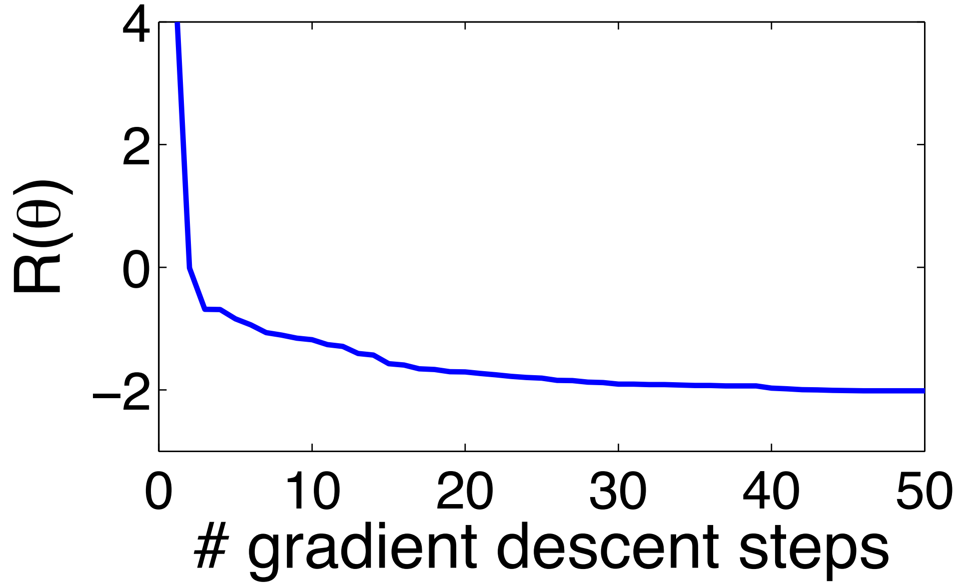
Experiment 1: Crossing lines

- We then attempt to use "DDD" to *preserve* discriminability of Task A, while *suppressing* discriminability of Task B.
- As the filter function *F*, we will use 2-D convolution, i.e., $F(\theta, x) = \theta * x$.
    - The filter parameter θ is the convolution kernel, which will be initialized to a 5x5 matrix sampled from $U[0,1)$.



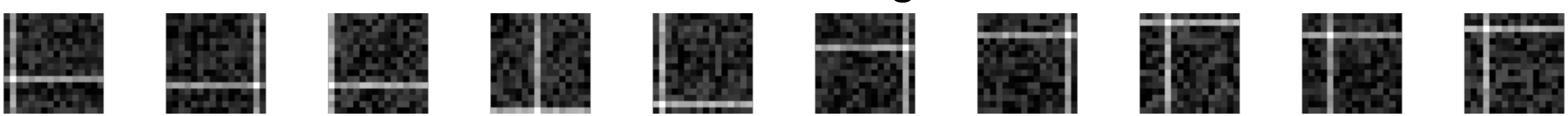
Experiment 1: Crossing lines

Gradient descent over θ

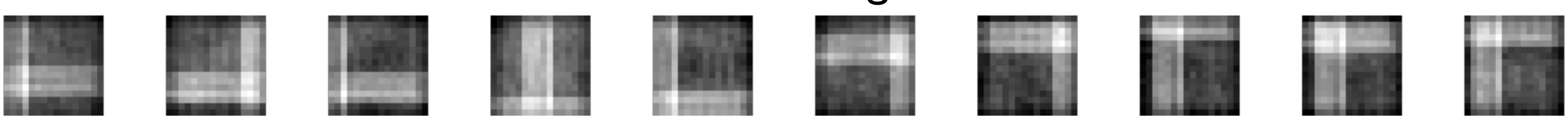


θ after 0 gradient descent steps.

Unfiltered images

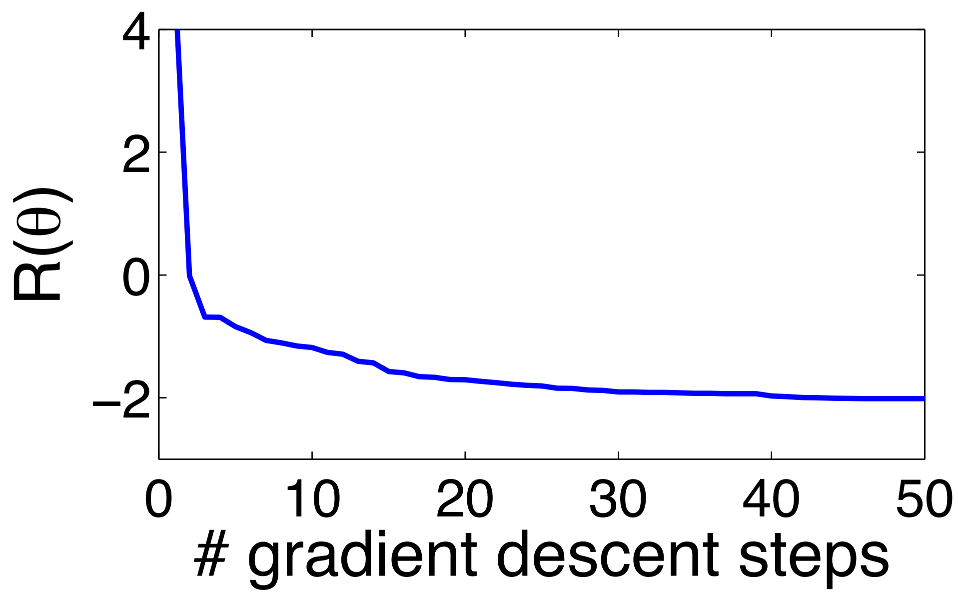


Filtered images

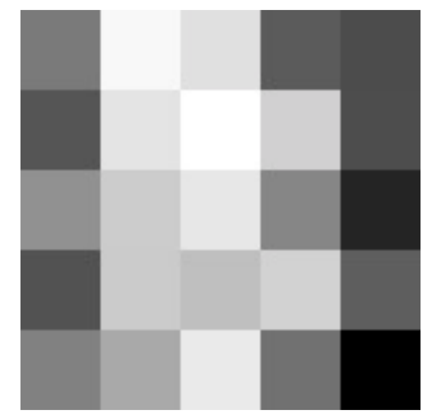


Experiment 1: Crossing lines

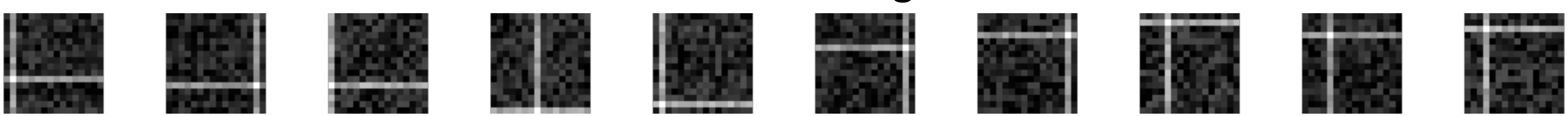
Gradient descent over θ



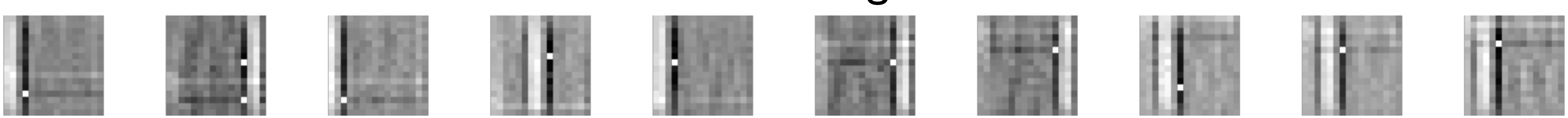
θ after 3 gradient descent steps.



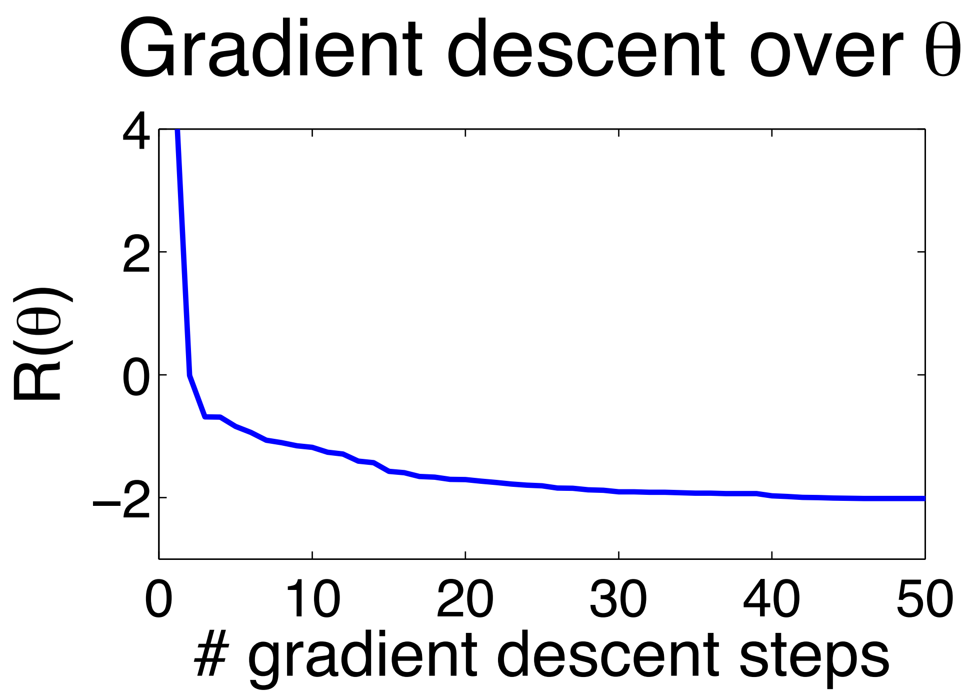
Unfiltered images



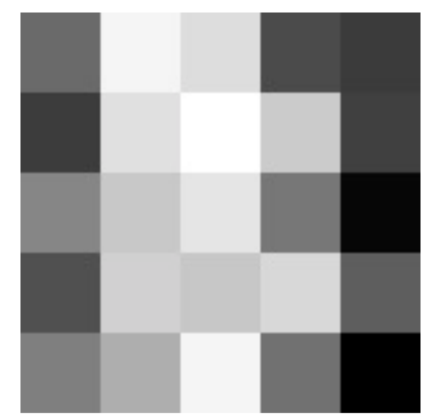
Filtered images



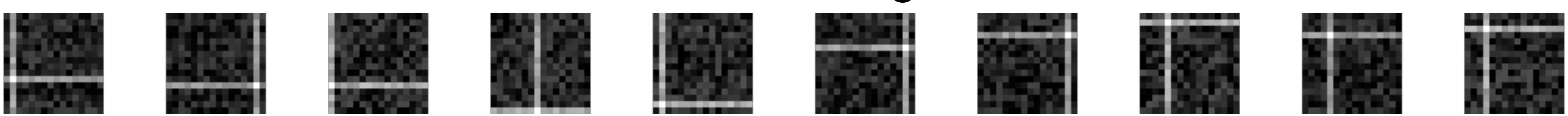
Experiment 1: Crossing lines



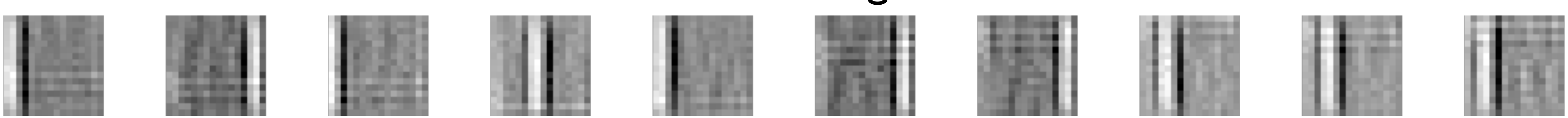
θ after 5 gradient descent steps.



Unfiltered images

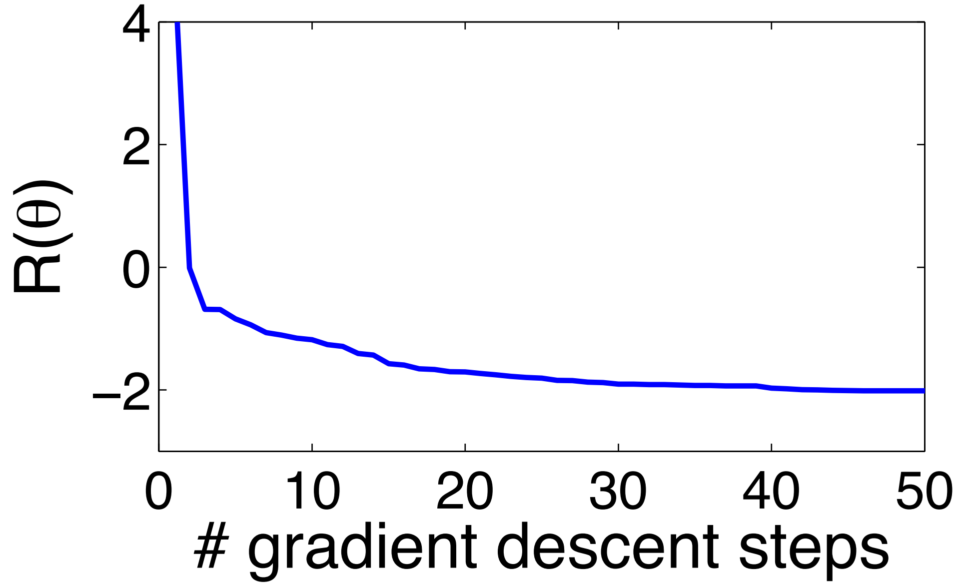


Filtered images

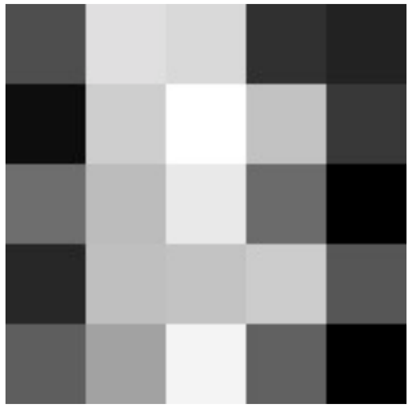


Experiment 1: Crossing lines

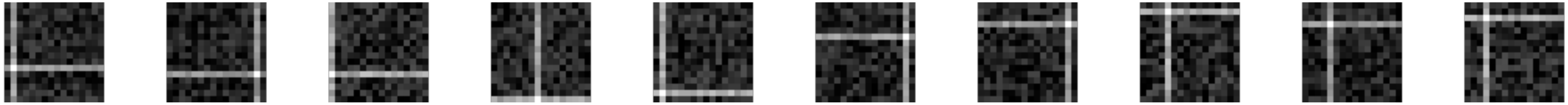
Gradient descent over θ



θ after 10 gradient descent steps.



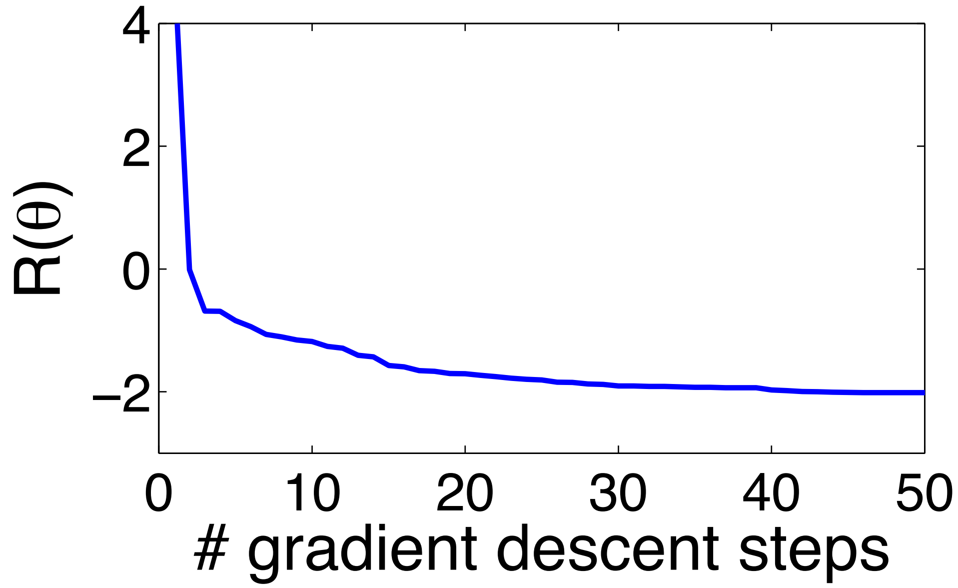
Unfiltered images



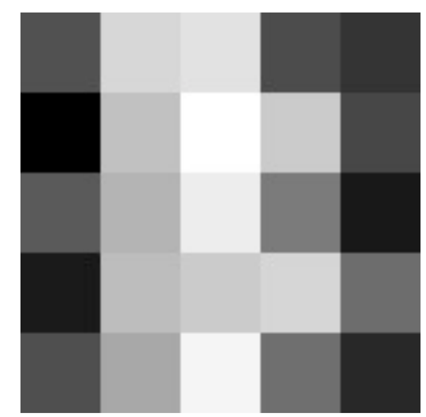
Filtered images

Experiment 1: Crossing lines

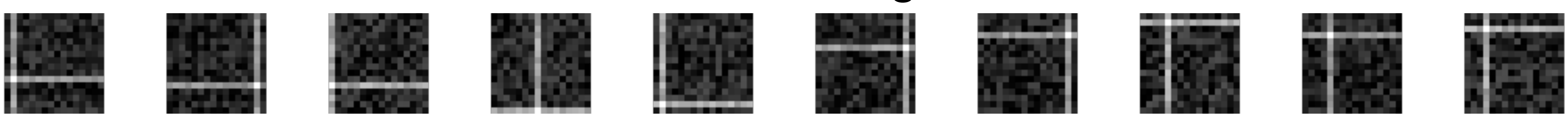
Gradient descent over θ



θ after 15 gradient descent steps.



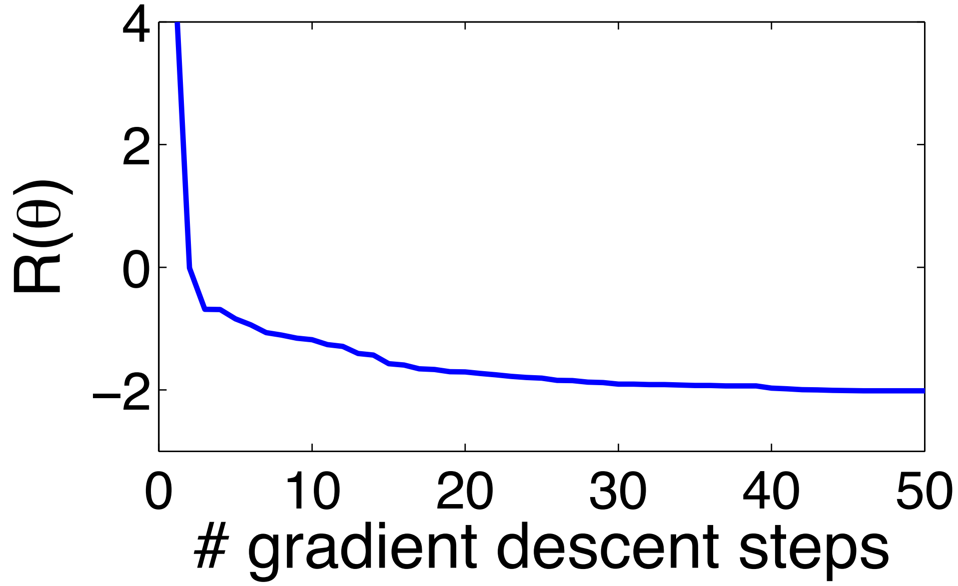
Unfiltered images



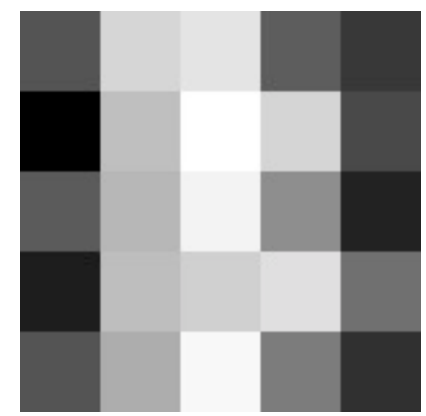
Filtered images

Experiment 1: Crossing lines

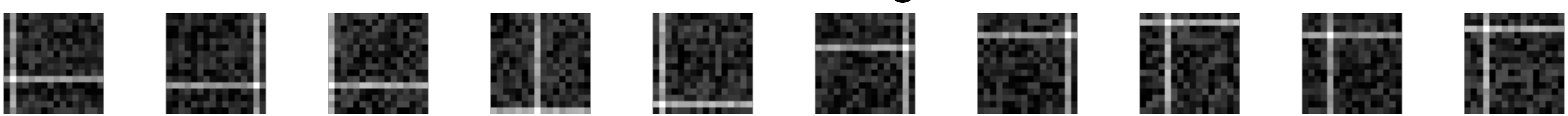
Gradient descent over θ



θ after 20 gradient descent steps.



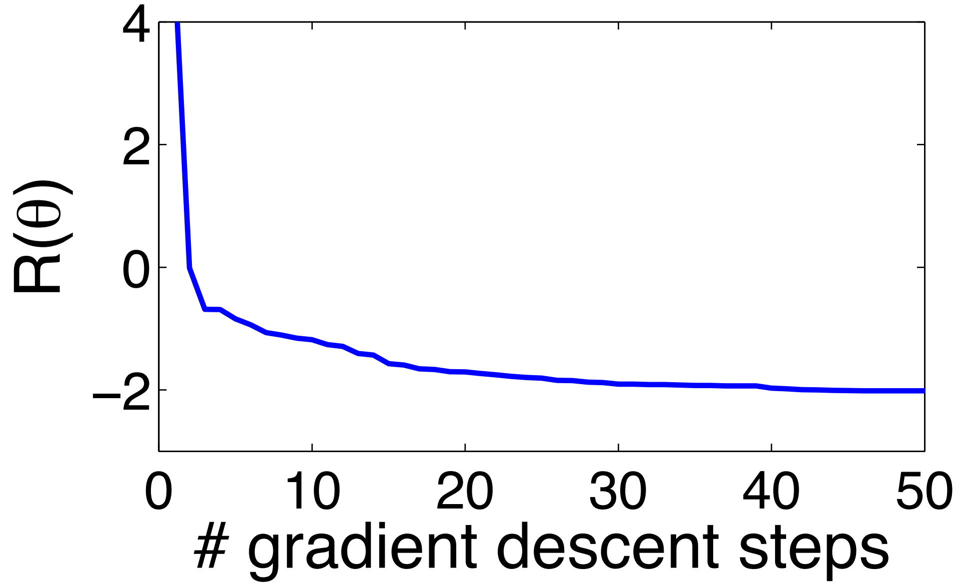
Unfiltered images



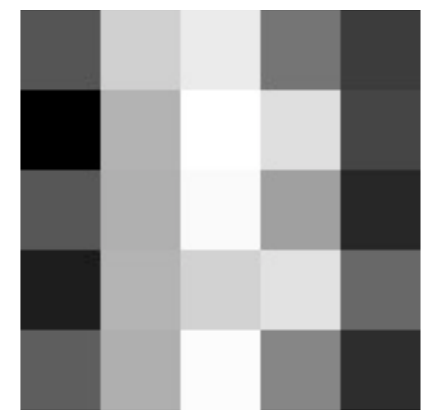
Filtered images

Experiment 1: Crossing lines

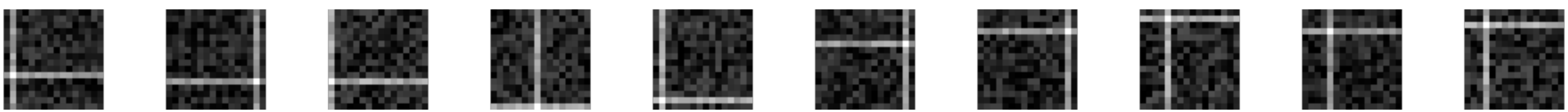
Gradient descent over θ



θ after 25 gradient descent steps.



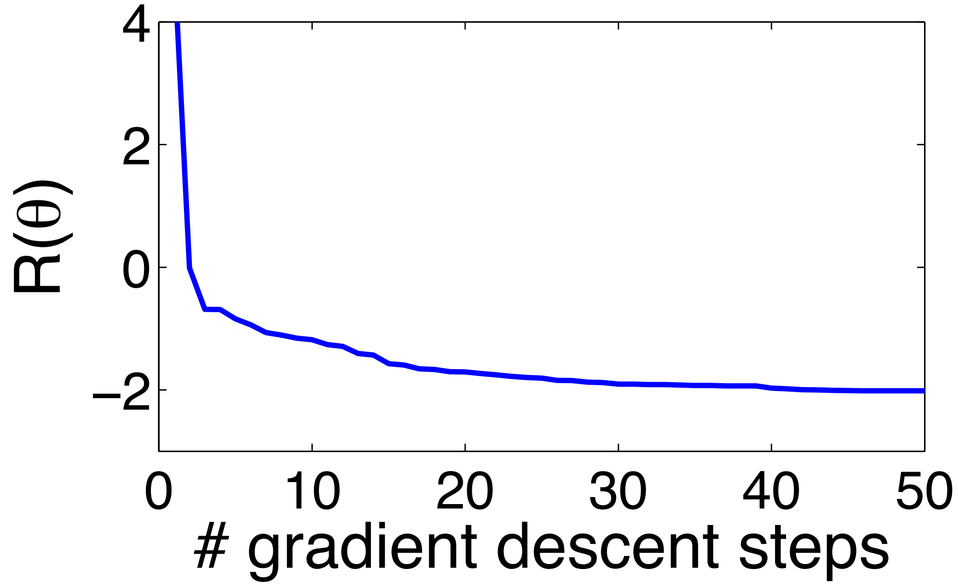
Unfiltered images



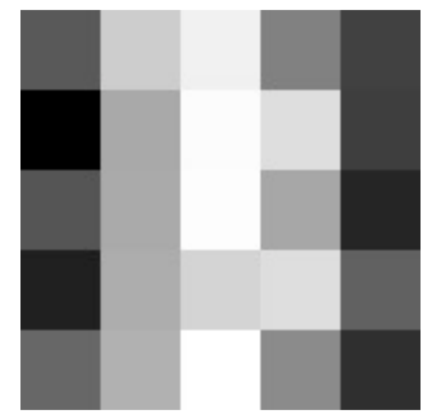
Filtered images

Experiment 1: Crossing lines

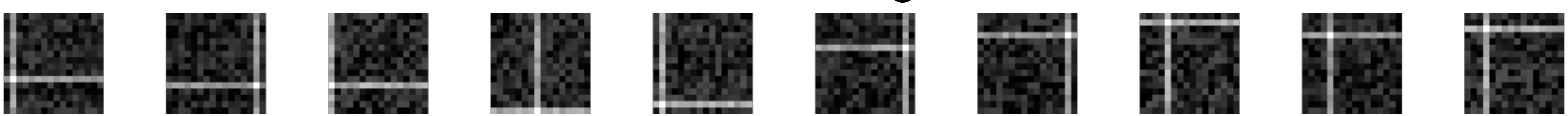
Gradient descent over θ



θ after 30 gradient descent steps.



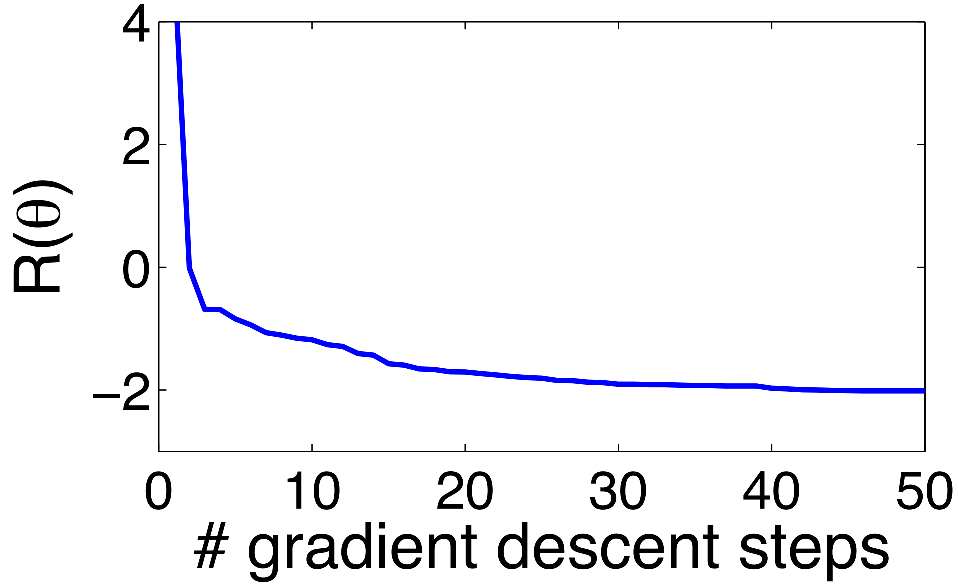
Unfiltered images



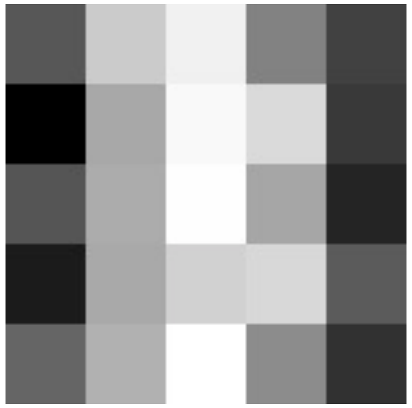
Filtered images

Experiment 1: Crossing lines

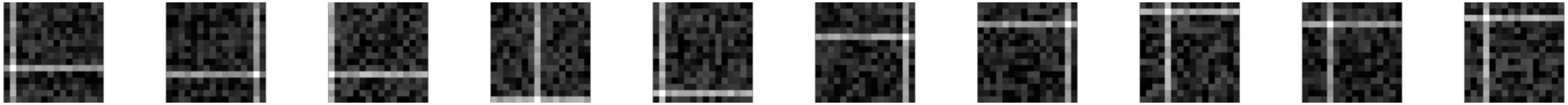
Gradient descent over θ



θ after 35 gradient descent steps.



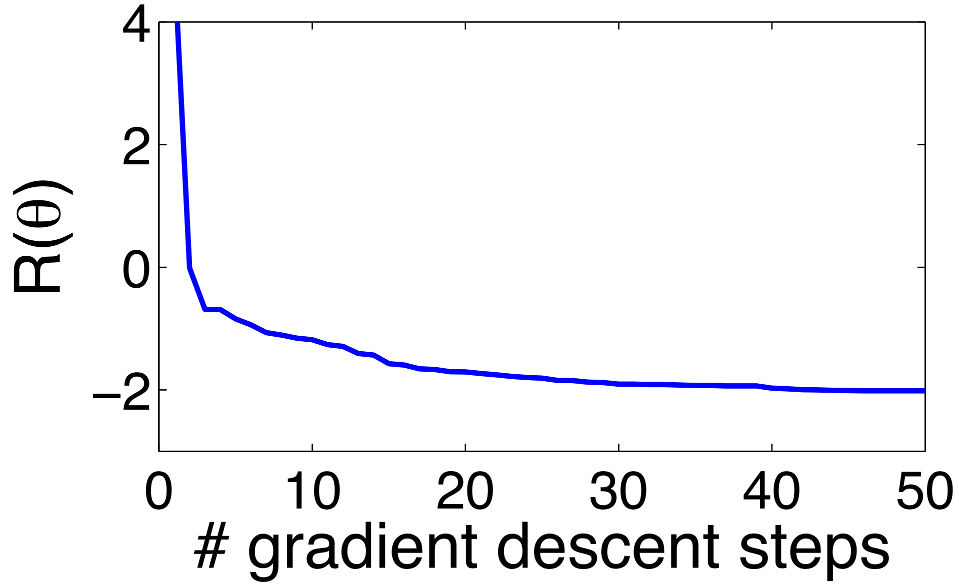
Unfiltered images



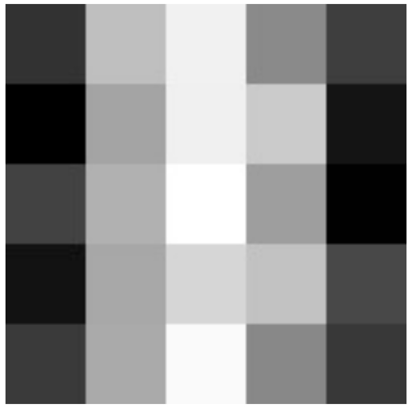
Filtered images

Experiment 1: Crossing lines

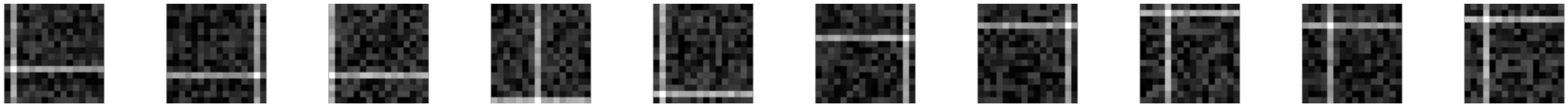
Gradient descent over θ



θ after 40 gradient descent steps.



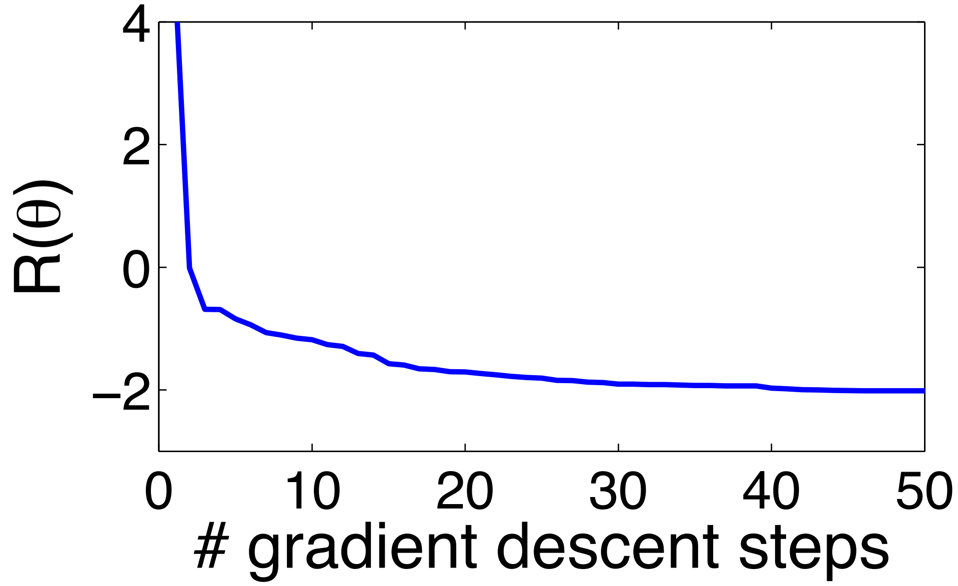
Unfiltered images



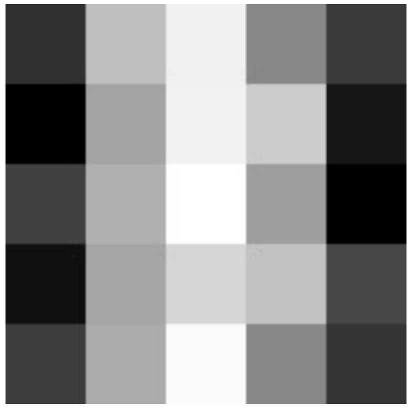
Filtered images

Experiment 1: Crossing lines

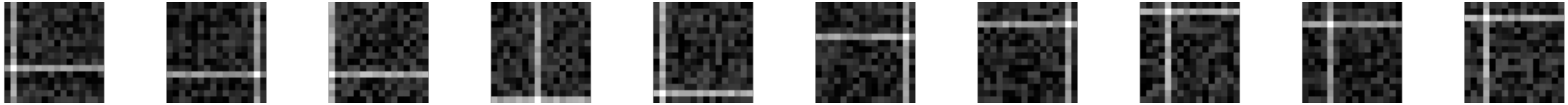
Gradient descent over θ



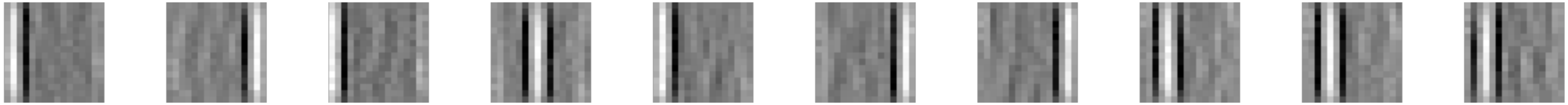
θ after 45 gradient descent steps.



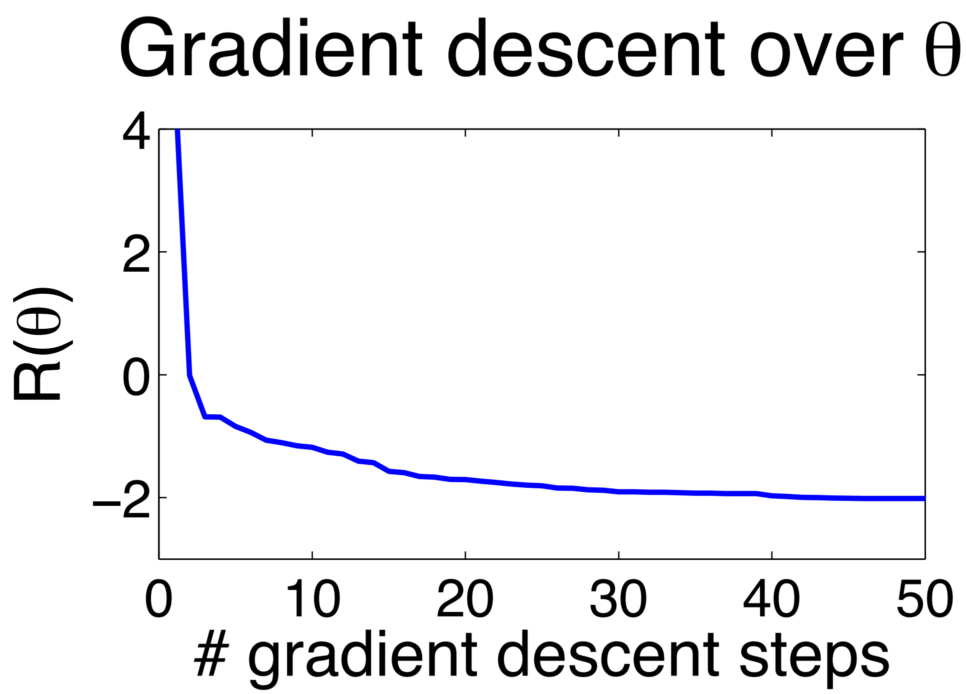
Unfiltered images



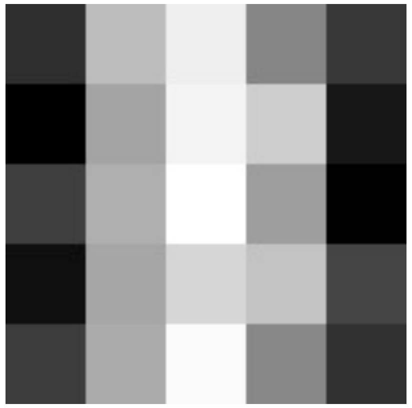
Filtered images



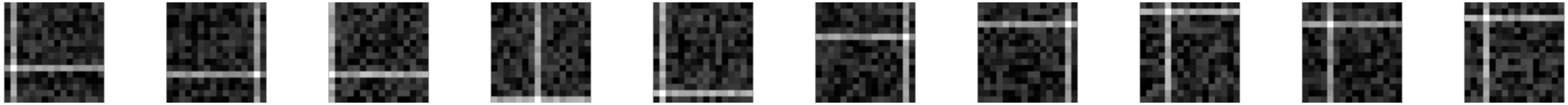
Experiment 1: Crossing lines



θ after 50 gradient descent steps.



Unfiltered images



Filtered images

Experiment 2: Preserve smile, suppress gender

- In our second experiment, we applied the "DDD" algorithm to the GENKI dataset, consisting of thousands of face images downloaded from the Web.

- GENKI was used to train a commercial smile detector.
- Images have been labeled for smile, gender, age, glasses, and more.

Experiment 2: Preserve smile, suppress gender

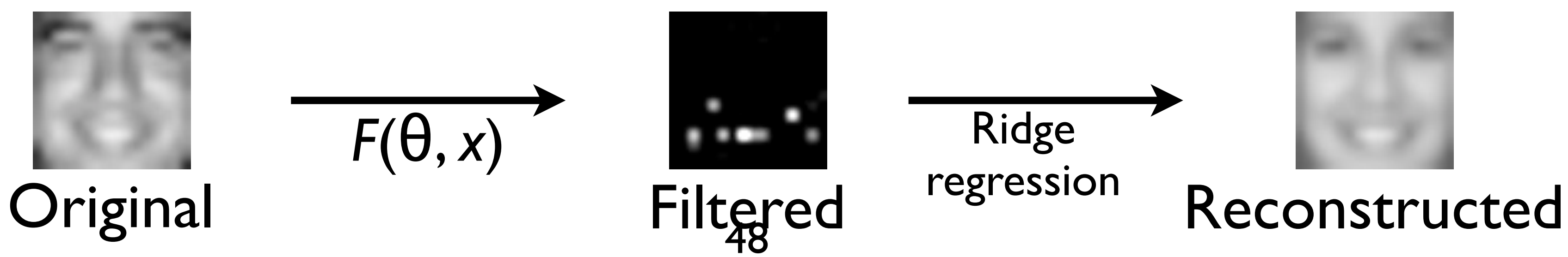
- We assessed whether "DDD" procedure could:
  - *Preserve* discriminability of *smile*, while *decreasing* discriminability of *gender*.
- Discriminability was assessed by uploading filtered images and querying human labelers on the Mechanical Turk.
- We used a pixel-wise "mask" filter:
  - $F(\theta, x) = \theta x$
    where θ is a diagonal matrix whose jth diagonal entry modulates the intensity of jth pixel of image x.

Experiment 2: Preserve smile, suppress gender

- We selected 1740 frontal GENKI images (downscaled to 16x16 pixels):
  - 50% smile, 50% non-smile
  - 50% male, 50% female
- Pixel-wise mask θ was initialized to random values.
- Executed "DDD" procedure to obtain optimal filter θ to maximize discriminability of smile, minimize discriminability of gender.

Experiment 2: Preserve smile, suppress gender

- The filtered images $F(\theta, X)$ are highly distorted compared to original X so that a human would not recognize them as faces.
- Hence, we execute an additional "reconstruction" step:
  - Apply *linear ridge regression* to regress from filtered images back to original images.
  - Ridge term ensures that only the "more discriminable" aspects of image are fully restored.
    - "DDD" property is maintained.

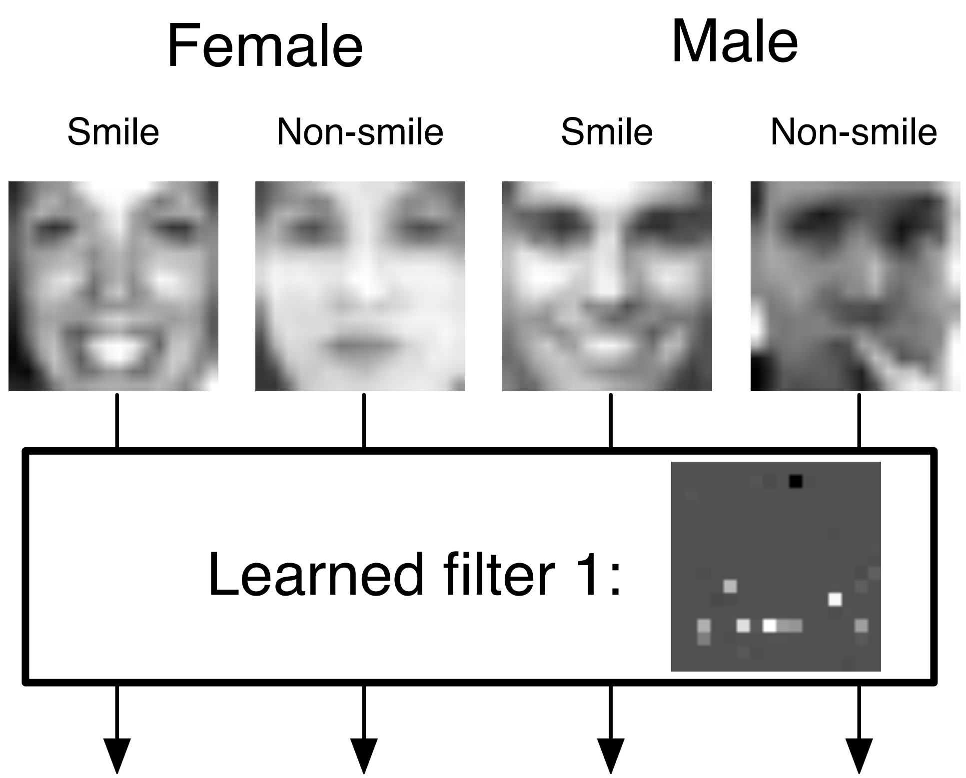


Experiment 2:
Preserve smile, suppress gender

- The "DDD" procedure learns the filter illustrated below:

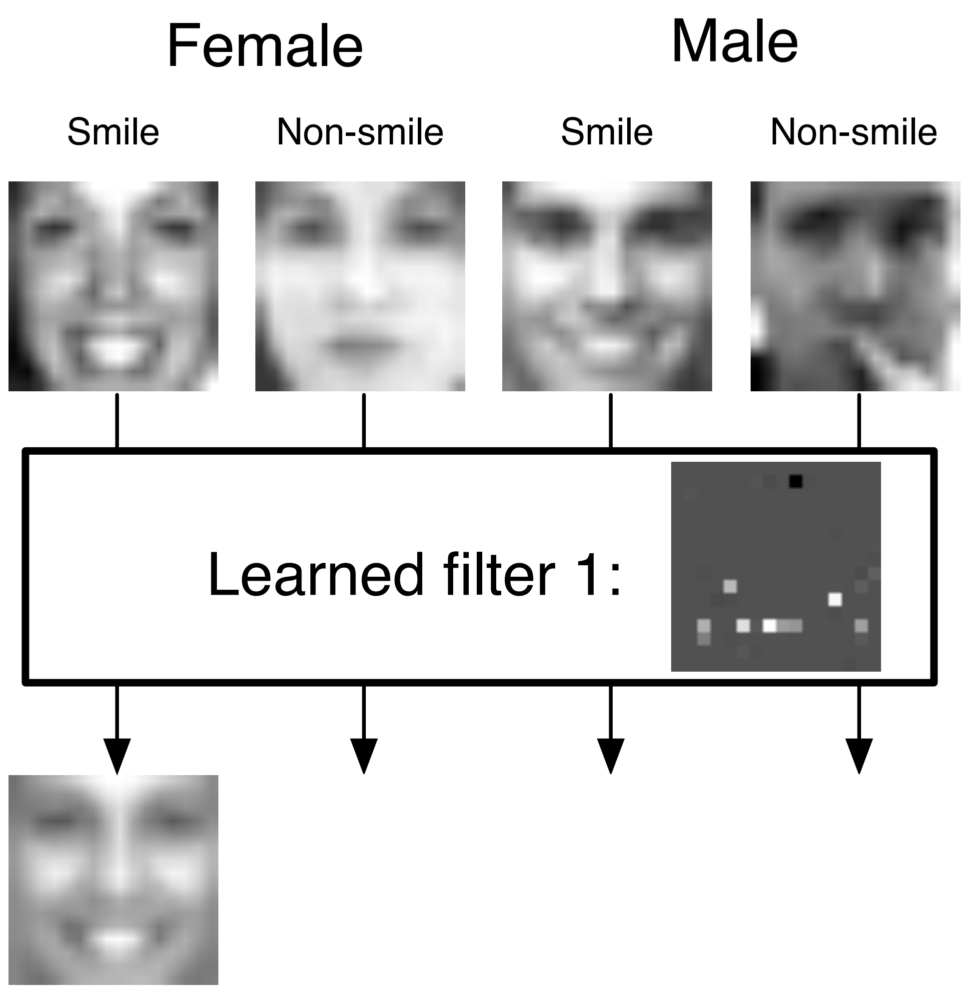
Experiment 2: Preserve smile, suppress gender

- The "DDD" procedure learns the filter illustrated below:



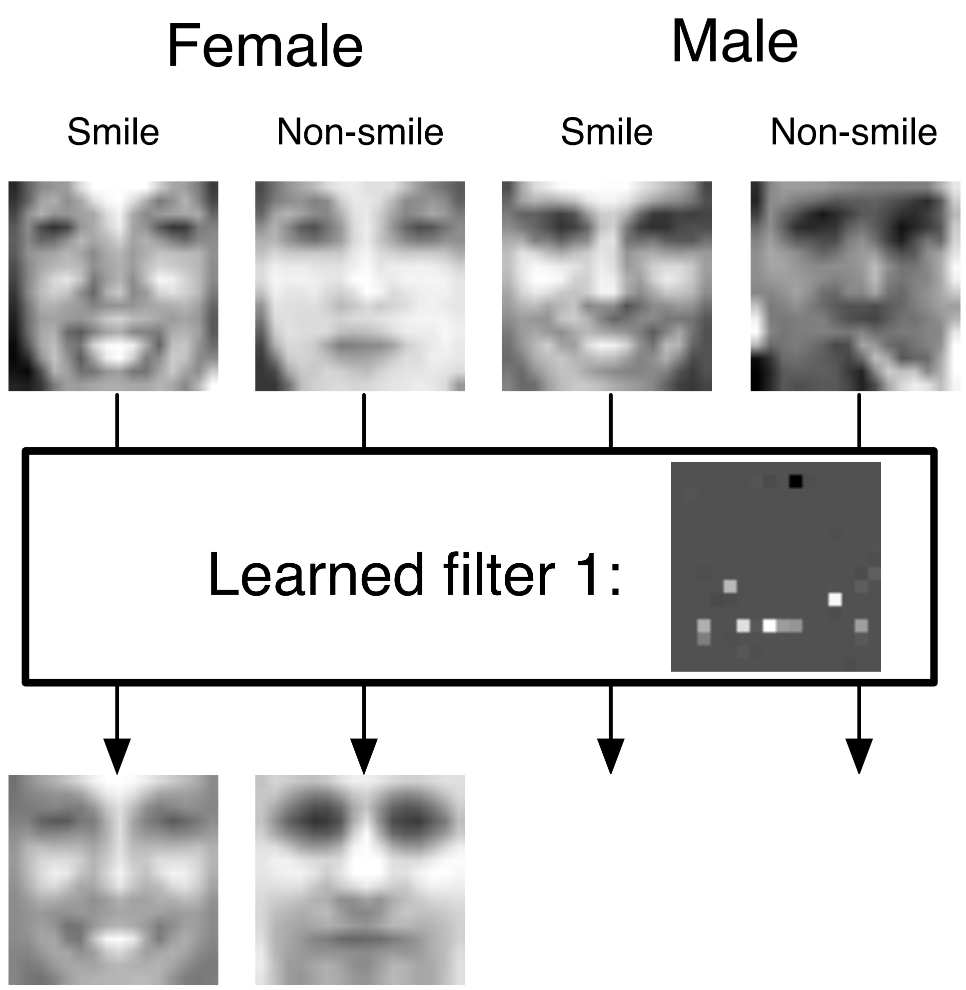
Experiment 2: Preserve smile, suppress gender

- The "DDD" procedure learns the filter illustrated below:



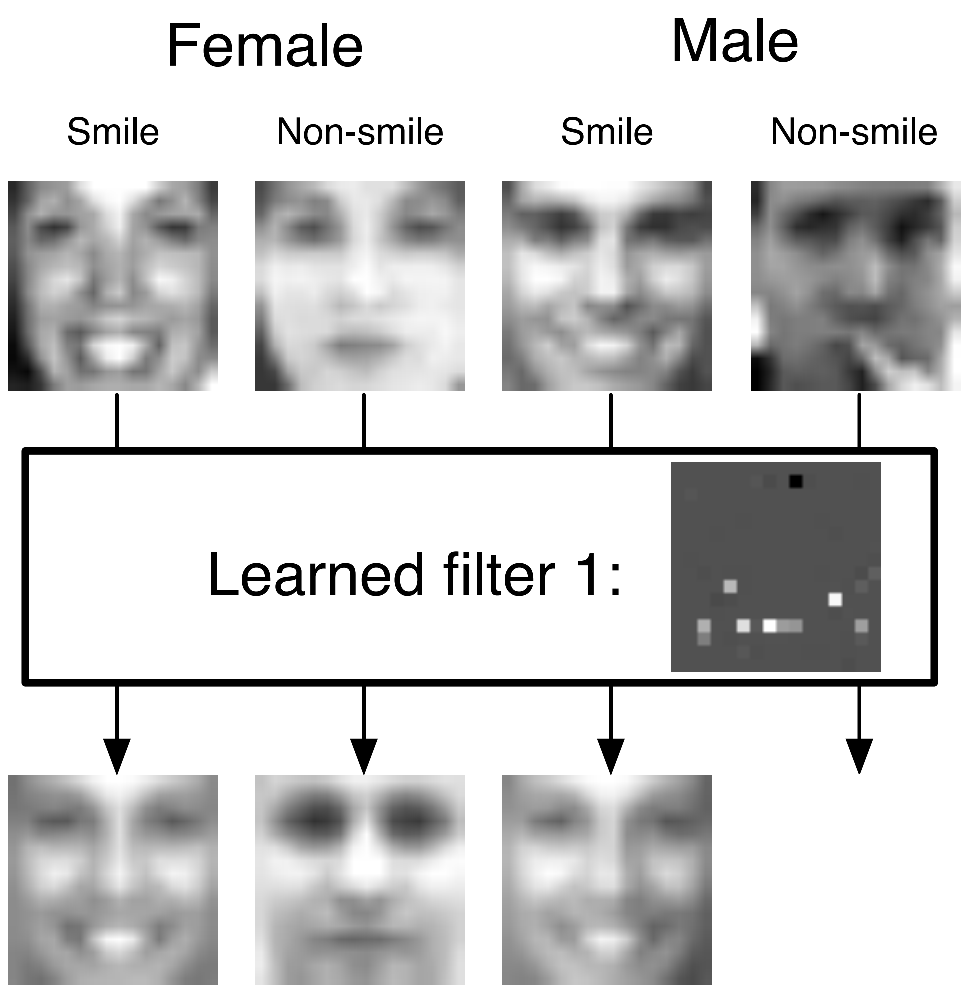
Experiment 2:
Preserve smile, suppress gender

- The “DDD” procedure learns the filter illustrated below:



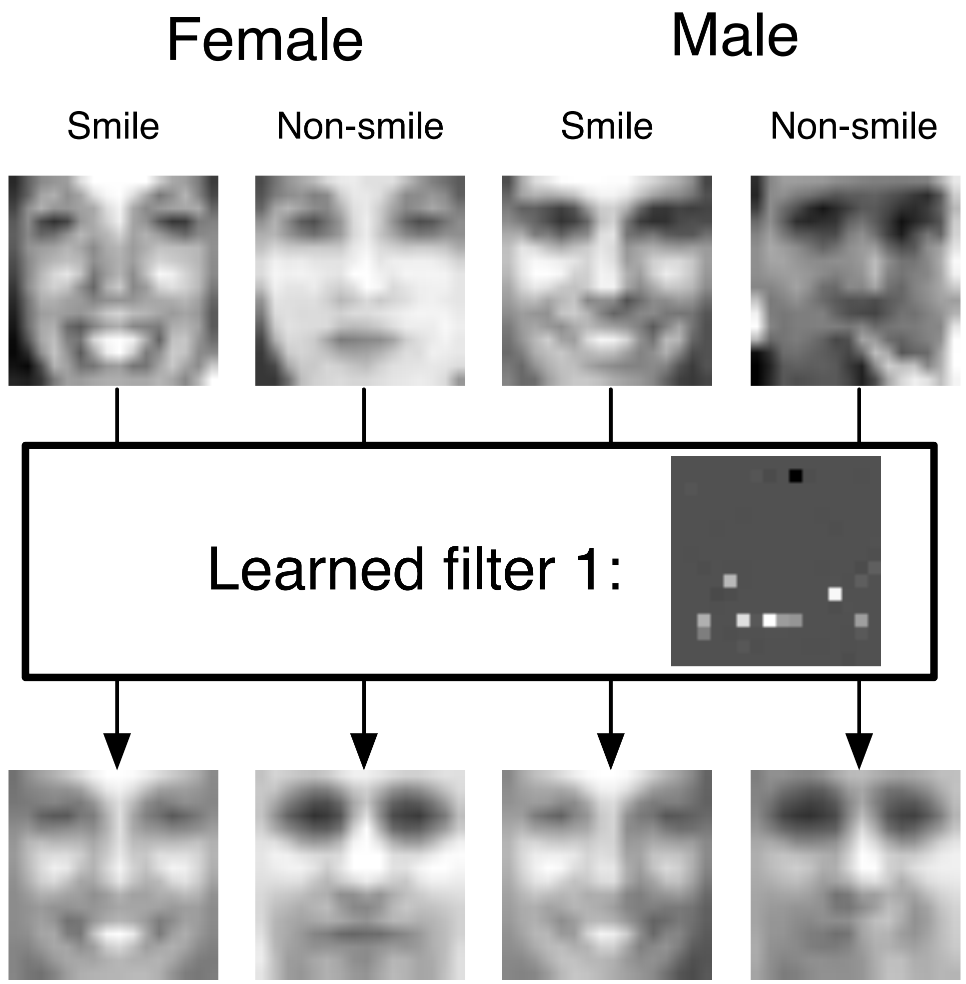
Experiment 2: Preserve smile, suppress gender

- The "DDD" procedure learns the filter illustrated below:



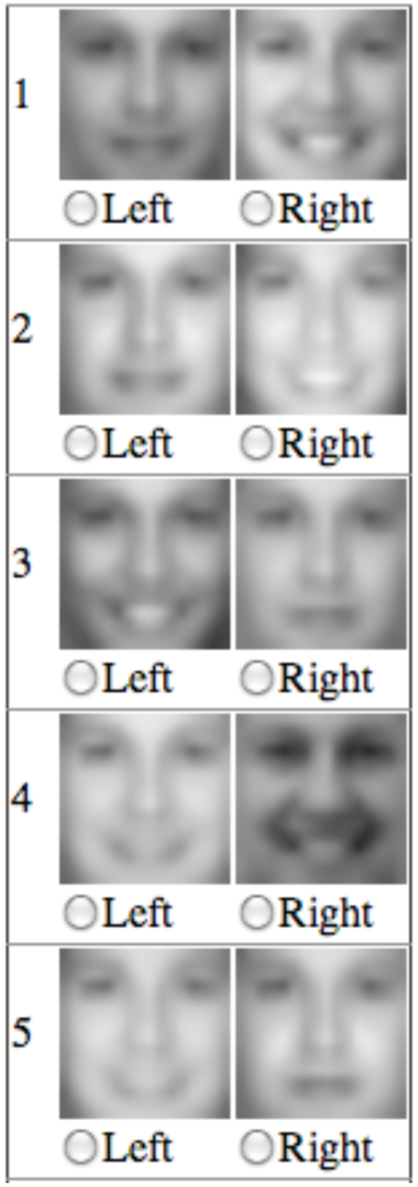
Experiment 2: Preserve smile, suppress gender

- The "DDD" procedure learns the filter illustrated below:



Experiment 2: Preserve smile, suppress gender

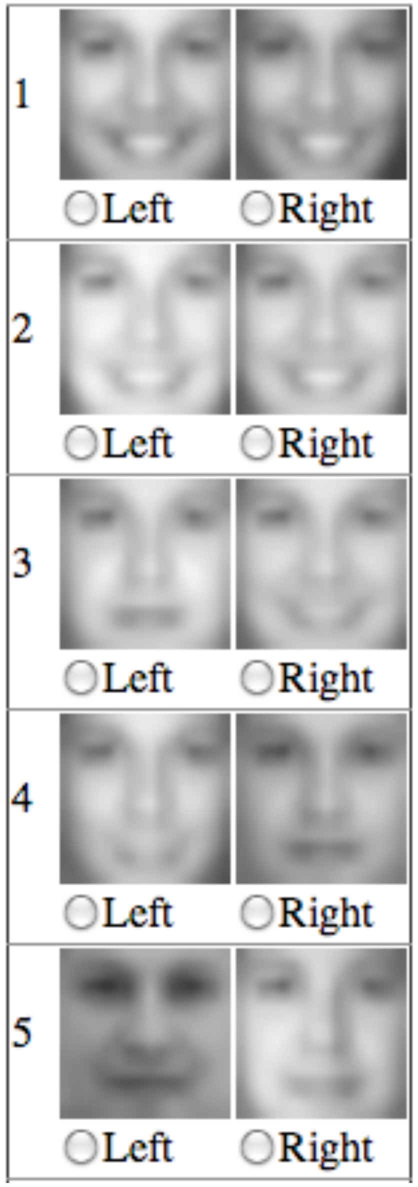
MTurk Task



- Using the learned preserve-smile, suppress-gender filter, we posted 50 pairs of *filtered* images -- 1 smiling, 1 non-smiling -- to the Mechanical Turk.
  - 10 Turk workers were asked to select which image of each pair was "smiling more".

Experiment 2:
Preserve smile, suppress gender

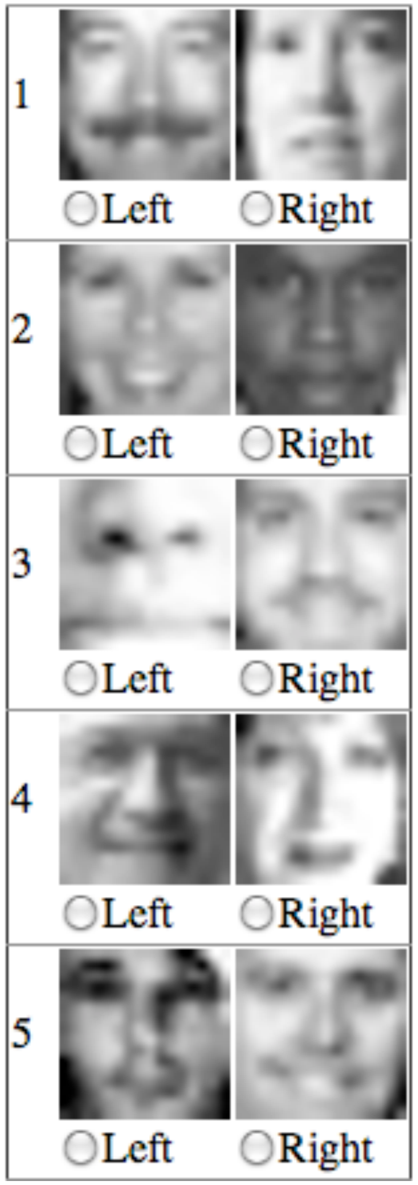
MTurk Task



- Using the same filter, we posted 50 pairs of *filtered* images -- 1 male, 1 female -- to the Mechanical Turk.
  - 10 Turk workers were asked to select which image of each pair was "more feminine".

Experiment 2: Preserve smile, suppress gender

MTurk Task



- Finally, for comparison, we posted 2 more MTurk tasks:
  - 50 smile/non-smile pairs of *unfiltered* images.
  - 50 male/female pairs of *unfiltered* images.

Experiment 2: Preserve smile, suppress gender

- Accuracy on each MTurk task was computed by taking majority vote across all 10 labelers for each pair.
- Results:

	Filtered	Unfiltered
Smile/non-smile	96%	94%
Male/female	58%	98%

Experiment 3: Hand-constructed filter

- For the task of preserving smile/non-smile discriminability, we could easily construct a filter by hand:
  - Only show the "mouth region" of each face.
- How well does this work compared to the filter learned using "DDD"?
  - We posted another MTurk task to test this.

Experiment 3: Hand-constructed filter

- It turns out that this manually-constructed filter allows considerable “male/female” information to pass through.
    - Despite strong prior domain knowledge, the learned filter performs better than manually created one.
- Results:

	Learned filter	Manual filter
Smile/non-smile	96%	96%
Male/female	58%	74%

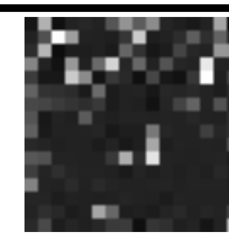
Experiment 4:
Preserve gender, suppress smile

- We also tried the *opposite* DDD task:
  - Preserve discriminability of *gender*, while decreasing discriminability of *smile*.
- We used exactly analogous procedures as for previous experiment.

Experiment 4: Preserve gender, suppress smile

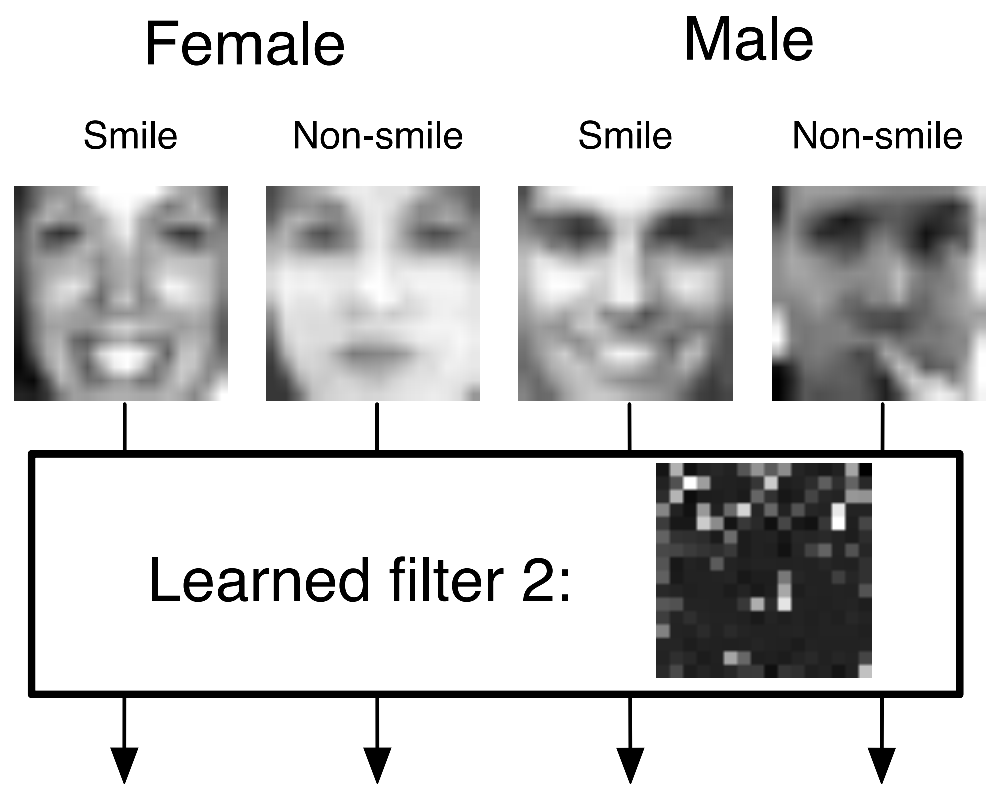
- Results:

Learned filter 2:



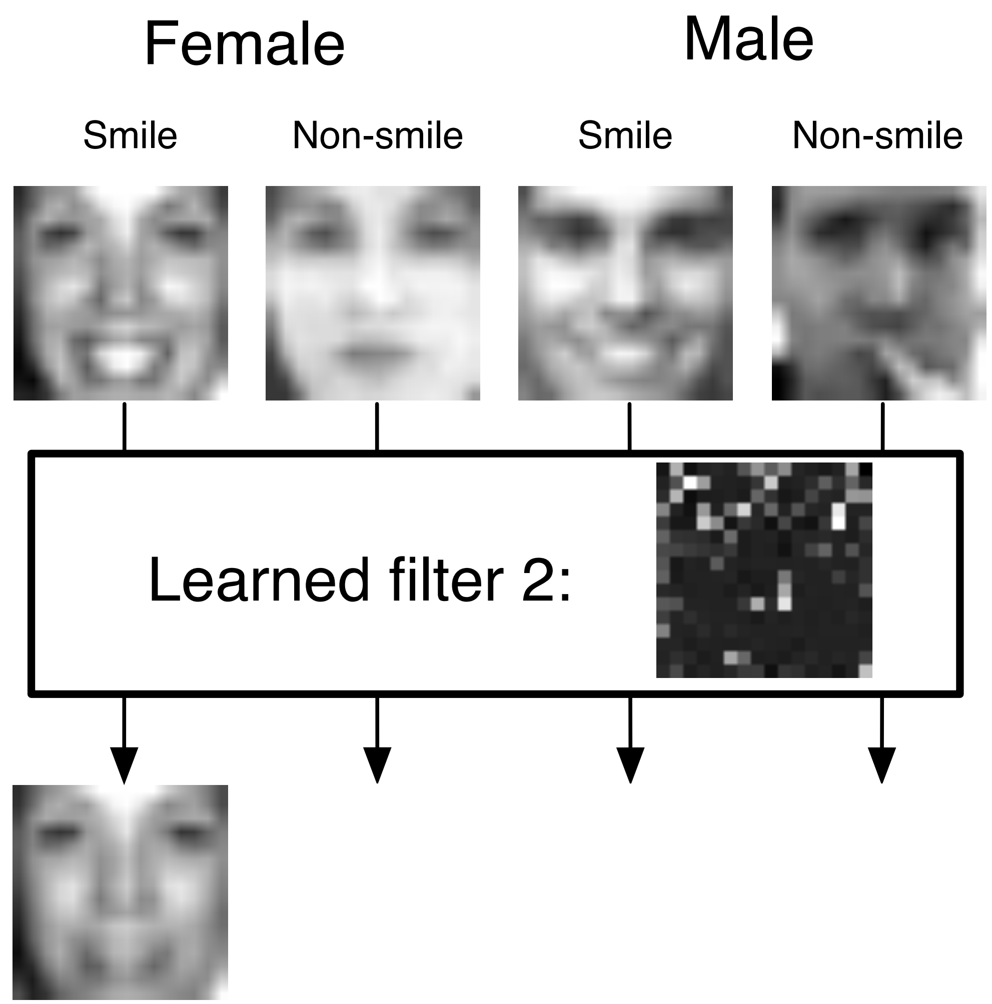
Experiment 4:
Preserve gender, suppress smile

- Results:



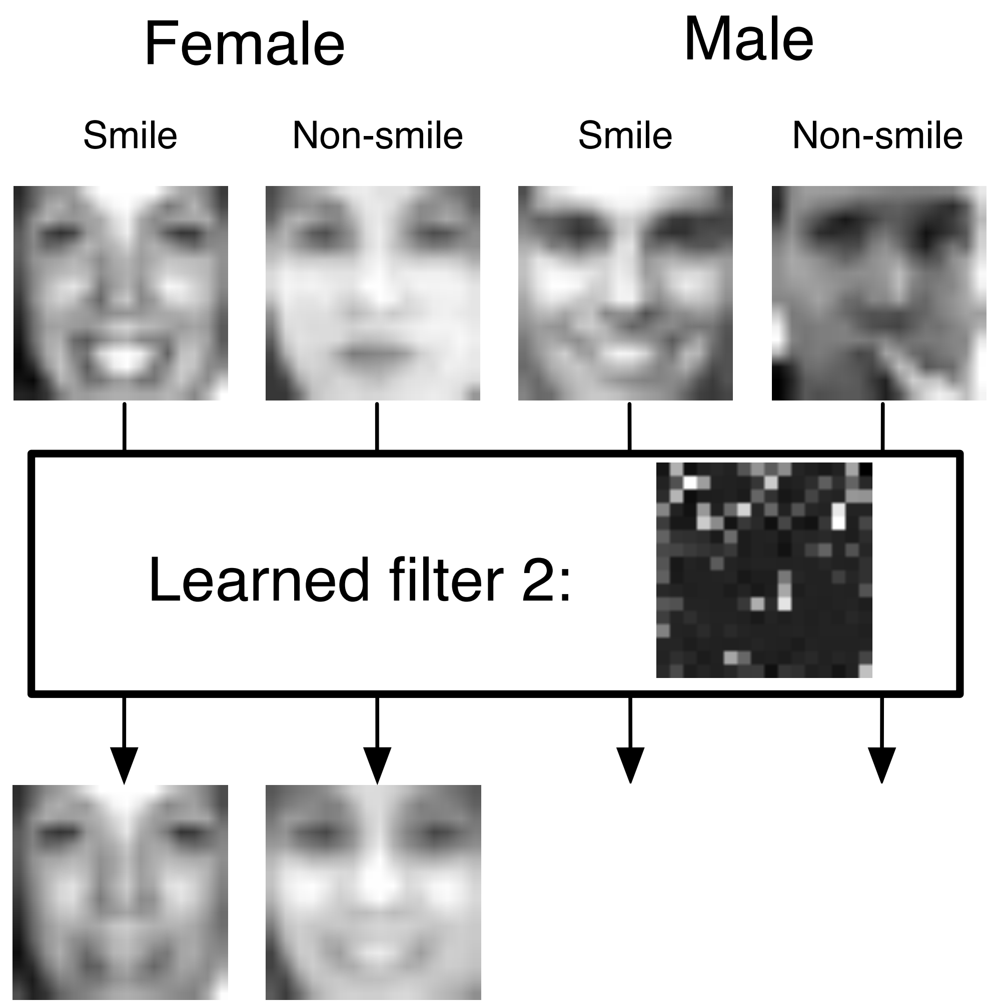
Experiment 4:
Preserve gender, suppress smile

- Results:



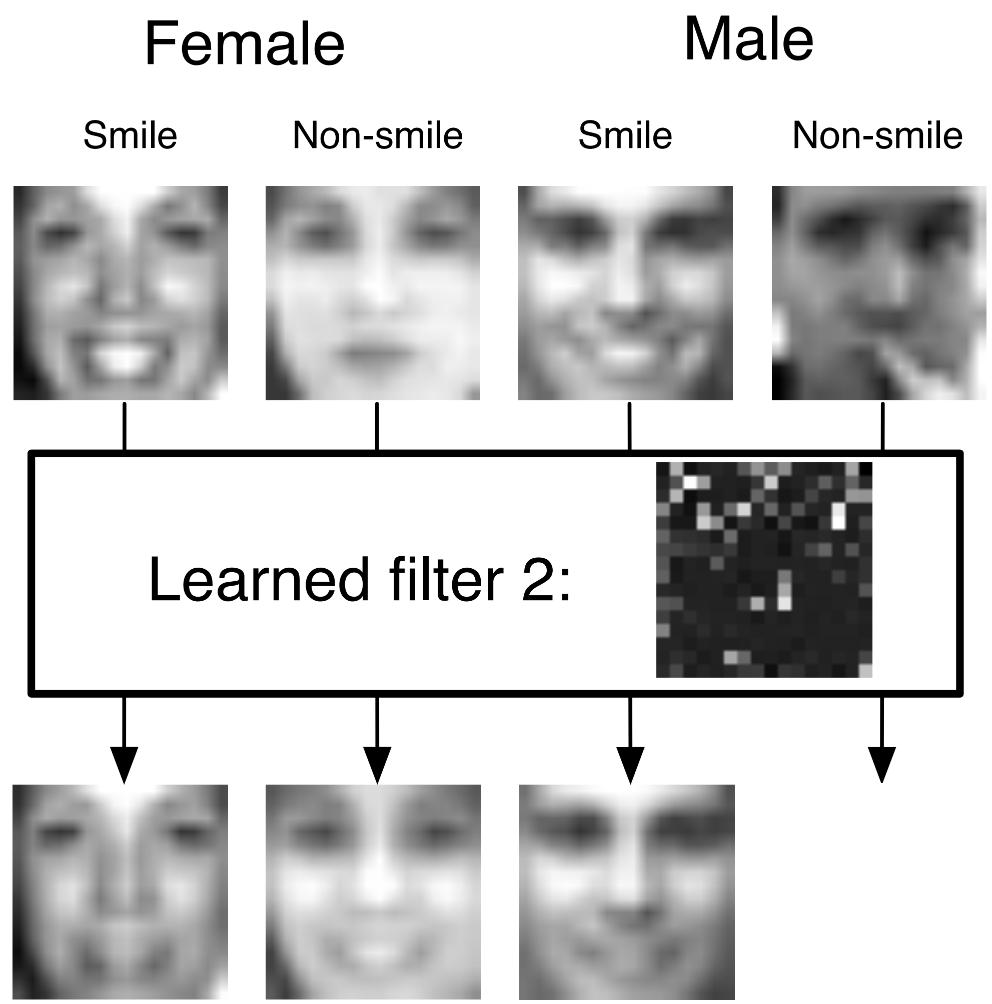
Experiment 4: Preserve gender, suppress smile

- Results:



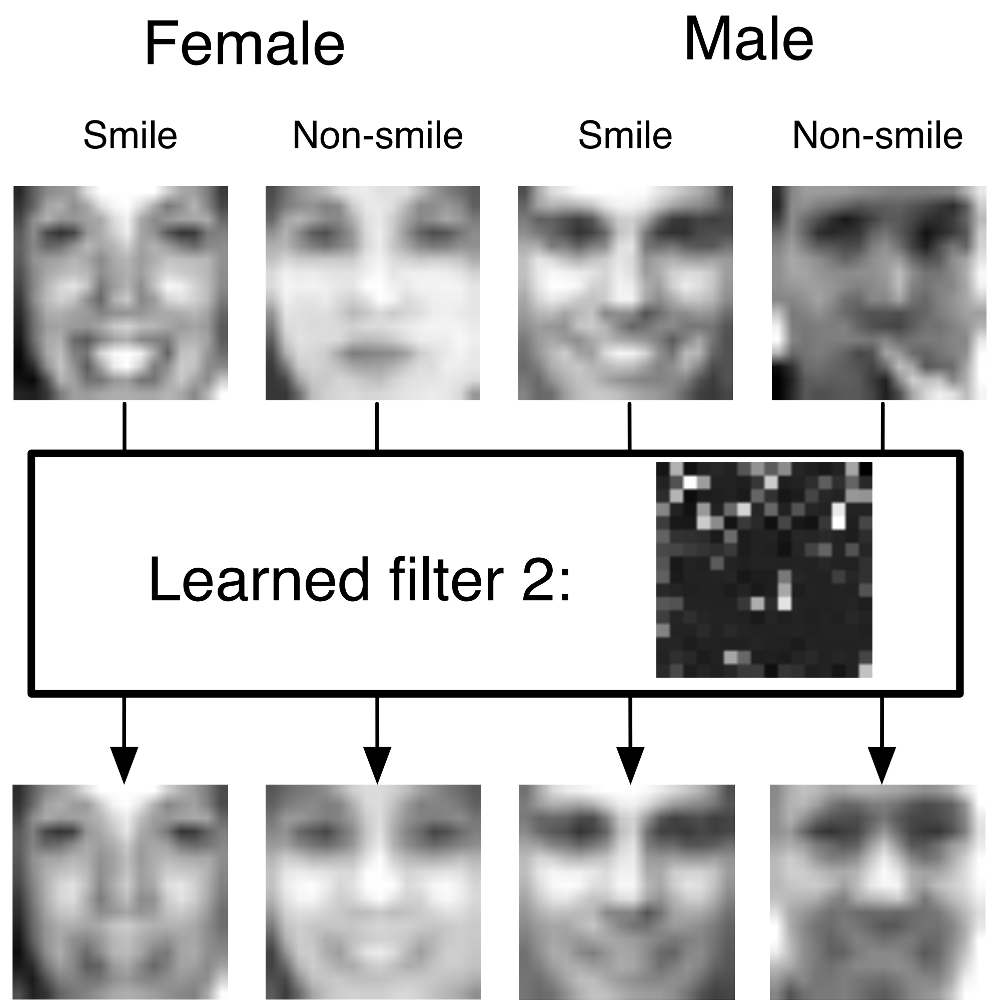
Experiment 4: Preserve gender, suppress smile

- Results:



Experiment 4: Preserve gender, suppress smile

- Results:



Experiment 4:
Preserve gender, suppress smile

- Accuracy on each MTurk task was computed by taking majority vote across all 10 labelers for each pair.
- Results:

	Filtered	Unfiltered
Smile/non-smile	64%	94%
Male/female	86%	98%

Suppression of facial identity

- As mentioned earlier, it would be useful to create a filter to preserve expression but suppress *facial identity*.
- In practice, we found that *suppressing gender* also removed considerable *identity* information.
- Consider the image below that was filtered with the preserve-smile, suppress-gender filter:
  - Which of the 10 faces below it is the unfiltered face?

filtered

a b c d e f g h i j

Suppression of facial identity

- To test efficacy of "face-deidentification" using "DDD" procedure, we created 20 face recognition questions:
  - Match *filtered* face to one of 10 *unfiltered* faces.
- To control for possibly "sloppy labelers", we randomly added 20 "control" questions:
  - Match *unfiltered* face to one of 10 *unfiltered* faces (this is trivial).
- The 10 labelers' responses were combined on each question using majority vote.

Suppression of facial identity

- Results:
  - Face recognition accuracy on filtered faces: 15% (guess rate = 10%)
  - Accuracy of best labeler on filtered faces: 30%
  - Face recognition accuracy on unfiltered faces: 100%
- Results suggest that suppression of gender also suppresses identity.

DDD for regularization

DDD for regularization

- The previous experiment described how DDD can be useful for *face de-identification* -- suppressing facial identity (via gender) while maintaining discriminability of expression.
- Another application of DDD is to partially *counteract covariate shift*.
- In this setting, we are more interested in *machine classification* of a "distractor" Task B (instead of human perception).

DDD for regularization

- Suppose we wish to train a classifier of attribute A using a training dataset D.
- Suppose that, in D, the attribute A is highly correlated with some other attribute B.
  - E.g., perhaps *smile* is strongly correlated with *gender*.
- If we apply the classifier trained on D to some other dataset in which corr(A, B) is different, then the classifier may perform very poorly.

DDD for regularization

- Using the "DDD" technique, we may be able to partially counteract this problem by *suppressing discriminability of B* prior to training the classifier for A.
  - In this case, DDD acts as a "application-specific regularizer" to ensure *invariance* to attribute B.
- Procedure for "regularizing" a training set using DDD:

1. Label training set for both A and B.
2. Learn filter θ using DDD to preserve A and suppress B.
3. Apply filter θ to training set.
4. Train classifier.
5. To classify a novel image, first filter it using θ, then classify.

Proof-of-concept experiment

- As a simple "proof-of-concept" experiment, we subsampled 4062 GENKI training images so that $\mathrm{corr}_{\mathrm{train}}$(smile, gender) = +0.64
- We also selected a disjoint test set containing 970 images for which $\mathrm{corr}_{\mathrm{test}}$(smile, gender) = -1
- We then trained two SVMs (RBF kernels) to classify an image as smile/non-smile:

1. SVM *with* DDD-regularization (filter was optimized to preserve smile, suppress gender)
2. SVM *without* regularization (classify unfiltered images).

Proof-of-concept experiment

- We then evaluated the trained SVMs on the *test set*.
- Results:
    - The "unregularized" SVM suffered due to the correlation between smile and gender on the training set:
        - Accuracy = 0.79 (area under ROC curve)
    - In contrast, the "regularized" SVM (using filter learned by DDD) was somewhat invariant to this correlation:
        - Accuracy = 0.92 (area under ROC curve)

Summary

- The proposed "DDD" algorithm can learn filters to preserve an attribute A while suppressing an attribute B.
  - Requirements: discriminability metric *D*, and filter function *F*, are available in *closed form*.
    - We used "maximal Fisher discriminability" for *D* -- other choices may work too.
- DDD can help to "de-identify" frontal face images while preserving their facial expression.
- In a proof-of-concept experiment, we illustrated how DDD can help to counteract covariate shift by providing invariance to specific image attributes (e.g., gender).

End