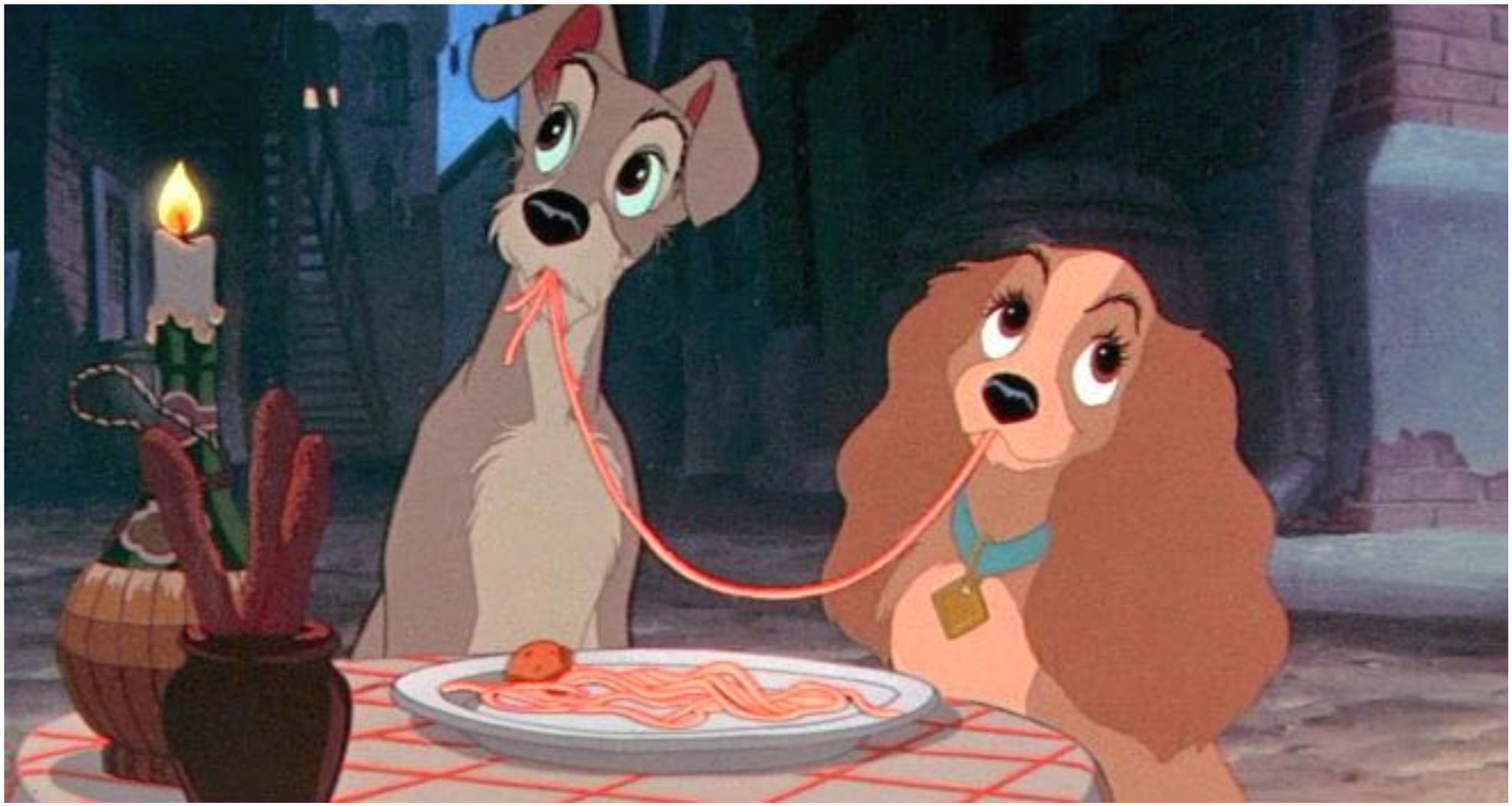
Connections between learning and pseudo randomness

Russell Impagliazzo, UCSD

We divide mathematics up into sub-fields for convenience

But artificial divisions hide deeper connections



But this partitioning process is arbitrary and can hide deeper

When do complex objects have simple ``models''?

- In science: finding mathematical models for complex phenomena

- In mathematics: finding simple approximations to graphs or other structures

- In learning: finding a simple generative model for random data points

Models in science

Scientists might be trying to understand the variety of butterflies within a forest

Limited ways of measuring

- Color: Azure, wingspan: 1.2 cm, antennae length: . 11 mm, proboscis length: 2.5 cm,…..

What would a mathematical model of butterflies entail?

- Model: A way to generate data points of the same format as measurements of a butterfly

- Strict realist: all variables in model should be meaningful in terms of butterflies

- Realist: Variables might not mean anything of themselves, but distribution produced should be close to true one

- Instrumentalist: Model might not be real, but predictions should be valid, i.e., model should be indistinguishable from the true one with respect to simple functions of the data

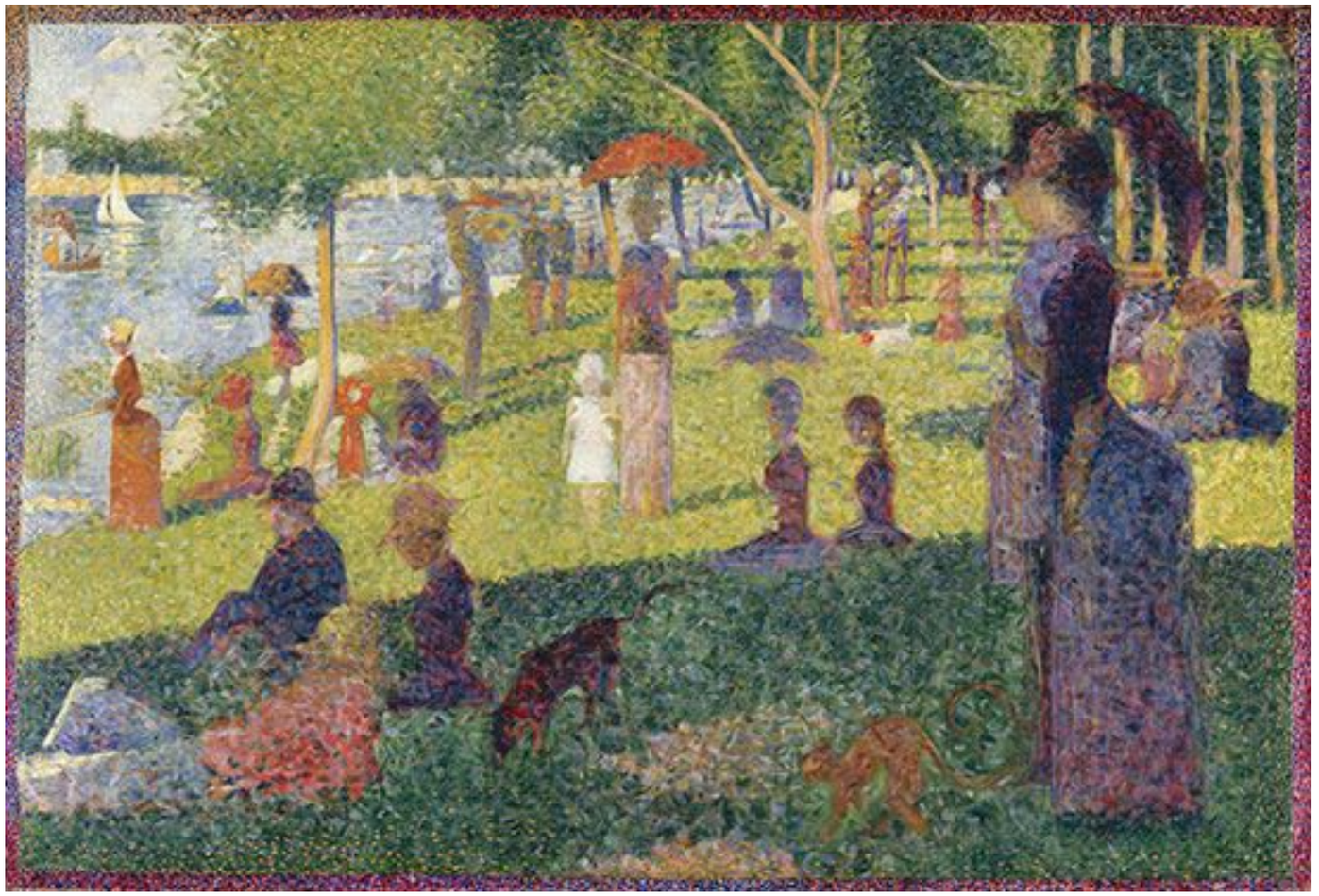
When do models exist?

- Model should be defined in terms of tests that can be made, simplicity means few tests actually involved in model

- Predictive power: Model should accurately predict values of tests, even if not used in model

- Realist: When can we come close to the original distribution? What does this even mean? (Is the distribution all butterflies we caught, all butterflies in the forest, or all butterflies that could exist?).

Many mathematical objects have structure at a high level



But the details appear random

Abstract setting

- We want to understand a probability distribution D over objects x

- There is a prior distribution U , often thought of as the uniform distribution on a set containing D's support

- We are interested in how Boolean tests T(x) from a class *T* behave on D, i.e., to estimate T[D]= Prob [T(x)=1] when x is drawn from D, for all T in *T*

- D might be complex, so we wish to find a ***model*** of D, a simple distribution D' so that D and D' are indistinguishable to within e by tests in *T,* i.e. |T(D)-T(D')| < e for all T in *T*

What is simple?

- Simple means definable in terms of relatively few tests, each from our class

- If G is a class of functions (not necessarily boolean), we use Gk[*T*] to be all functions of the form g(T1(x),…Tk(x)) where each Ti is in *T*

- A measure u is a function from U to [0,1], and induces the distribution Du where we pick x from U, and keep x with probability u(x), otherwise resampling. |u| = Exp_U [u(x)] is the ``density'' of u

Example settings

- U is the uniform distribution on a square , and each T is a straight line with true on one side, false on the other

- U is the uniform distribution on all n bit strings, and *T* is the class of tests with small circuit complexity

- U is the set of all integers up to N, and *T* is the class of Gowers norms of functions.

- U is the set of edges of the complete graph on N vertices, and each T is a cut, a pair of disjoint subsets A and B, where an T(e) is true if e has one endpoint in A and the other in B.

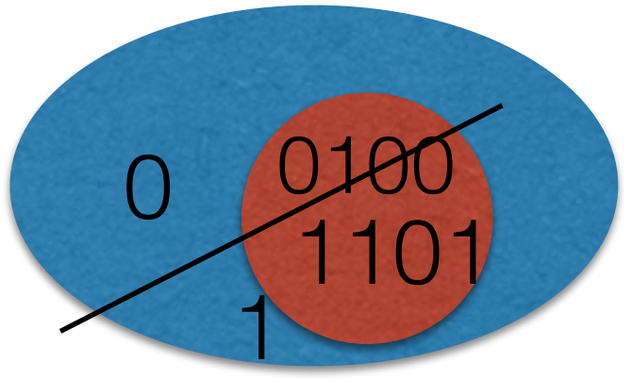
- U is the space of all images, and T is the class of tests defined computable by neural nets with a certain configuration

Pseudo-density

- A distribution is pseudo-random if it is indistinguishable from U
- For T a test, and D a distribution, let T[D] = Exp_{x chosen from D} [T(x)]
- A distribution has density > d if for all (measurable) sets S, S[U] > d S[D].
- D has pseudo-density > d with error e with respect to ***T*** if T[U] > d T[D] - e for all T in ***T.*** In other words, using tests in T, and relatively few samples, there is no way to tell that the density of D is not > d
- One way to be pseudo-dense is to be actually dense within a pseudo-random sub-distribution.

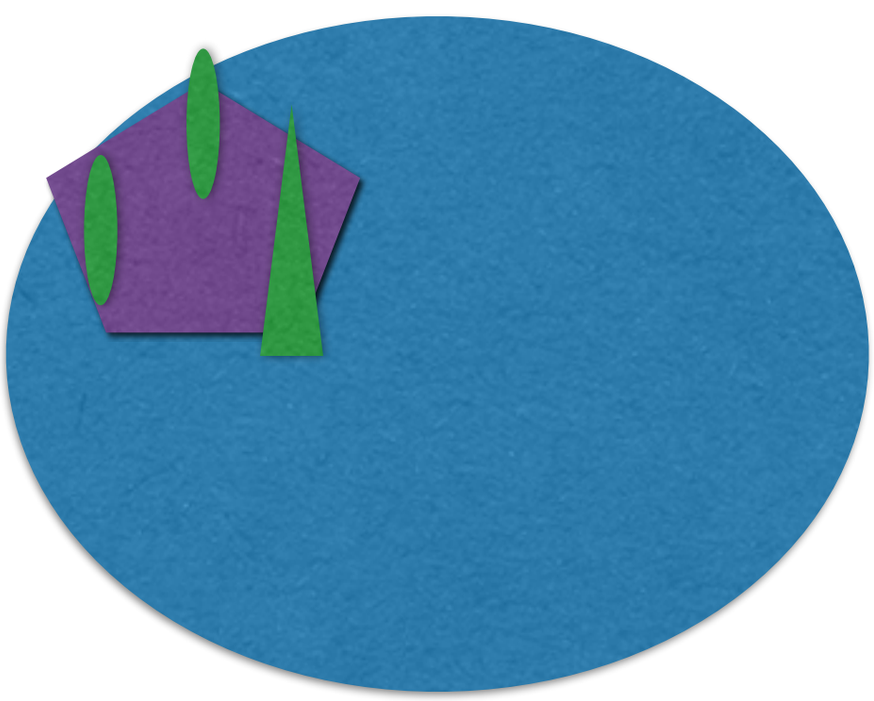
Three kinds of decomposition lemmas

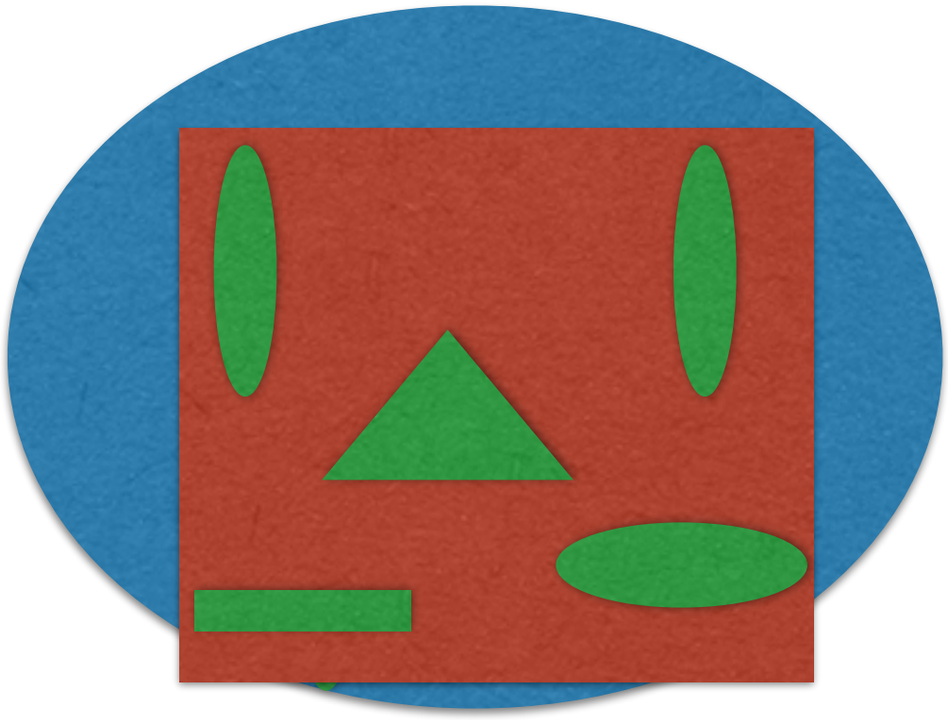
- Hard-core set lemmas [I, H: complexity]: Let f be any Boolean function on U. For some k = poly(1/e, 1/d), there is either a function F in Gk[*T*] so that Prob[f(x)=F(x)] > 1 -d or a distribution u in Hk[*T*, f] of density at least 2d so that f is pseudo-random on Du.



Transference Principles, aka Dense Model Theorem

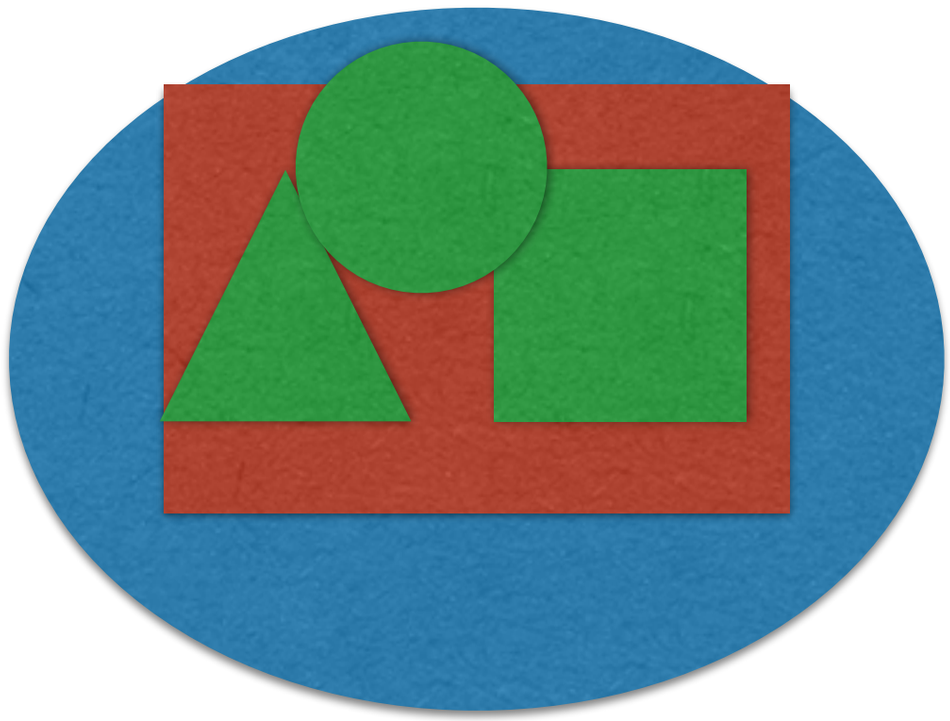
- Green-Tao,Tao-Ziegler , BSW, RTTV, here: If D is at least d pseudo-dense for Gk[*T*], then there is an indistinguishable distribution Du, which is at least d dense, {and u is in Hk[*T*] }





Regularity lemmas

- Szemeredi, FK, graph theory . If D is at least d dense, then there is a model Du with u in Hk[*T*]



Our results (but cf RTTV, TTV)

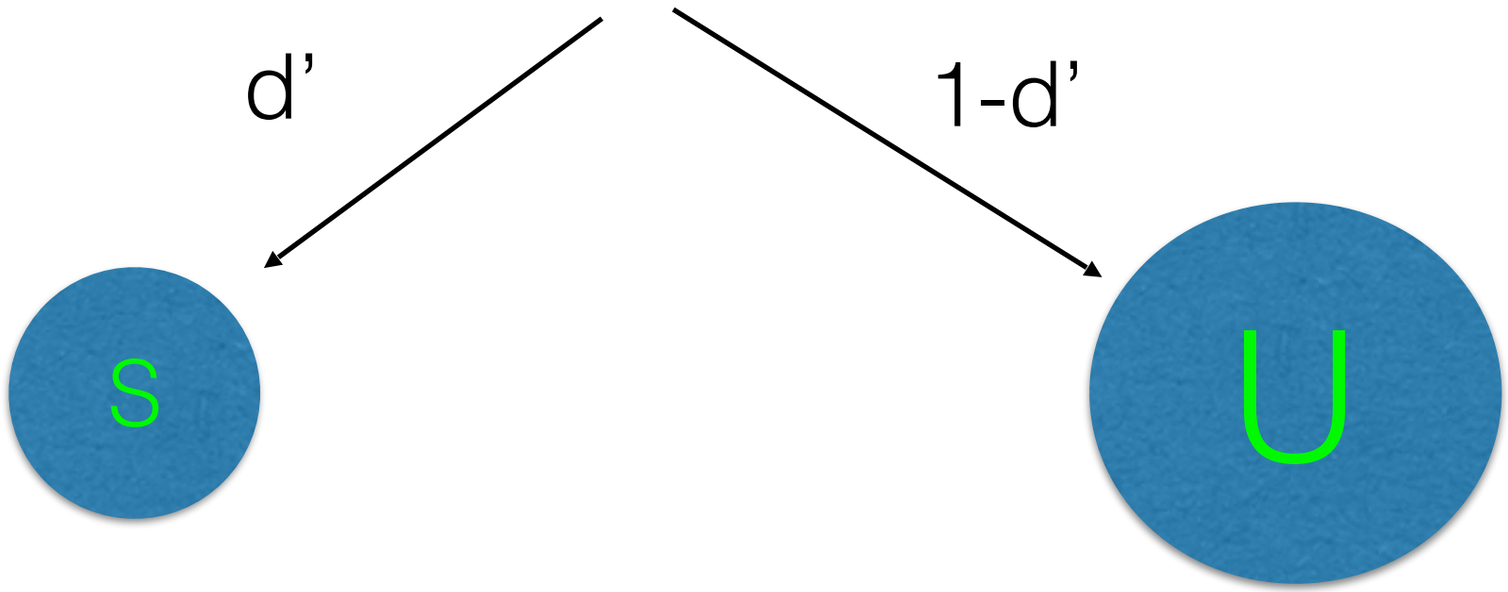
- Hardcore Set

Dense Model

Weak Regularity

By going through the reductions, get a variety of dense model lemmas and weak regularity lemmas (Different k, different G, different H)

Reduction from Hard-core to Dense Model



Counter-example to pseudo-density=
Computing f with error < d'
Model for S =
Hard-core distribution for f with density 2 d'

Algorithmic versions

- Some proofs of the Hardcore Set Lemma algorithmic, based on boosting algorithms (see KS)

- Boosting is a very successful technique from machine learning (Schapire, FS), putting together weakly correlated hypothesis into a single highly correct hypothesis

- Many different boosting algorithms known, most of which can be modified for constructive hard-core sets

Boosting

- Unknown function f on uniform distribution U
- Sub-routine: Weak Learner: Input: u_i , a measure defined in terms of f, h_1,..h_{i-1}. Output: a test h_i with correlation e with f on D_{u_i}
- Goal: quickly output a function H= G (h_1,..h_i) so that Prob_{x in U} [H(x)=f(x)] > 1 - d
- ``If weak learner always finds hi, boosting algorithm finds H in few iterations''

Reductions are algorithmic

- Each boosting algorithm with a density bound gives an algorithmic hard-core set lemma
- Each algorithmic hardcore set lemma gives an algorithmic dense model theorem : Need as a sub-routine a procedure for finding tests that distinguish between two distributions (that are not indistinguishable)
- Algorithm will either output a model u in Hk[*T*] of D or a test T in Gk[*T*] that is a counter-example to pseudo-density, i.e, $T(U) < d\ T(D) - e$.
- If D is actually dense, must produce a simple model, so also get algorithmic weak regularity lemmas.

GAN (Goodfellow et al)

- Possible tie back to machine learning. Generative adversarial networks use two learning algorithms to understand distribution. One (forger) tries to learn to produce samples from a distribution. Other (critic) tries to guess whether samples are genuine or forged.

Using Algorithmic DMT as GAN-type algorithm

- Generation through adversarial boosting (GAB) Strategy where either the critic finds a narrow criteria for genuine samples or the forger finds a high entropy way that fools the critic.

Using Algorithmic DMT as GAN-type algorithm

- Forger: Simulates boosting algorithm

- Critic: Distinguisher between forger's distributions and S = Weak learner.

- Forger produces candidate distribution u_i, Critic produces distinguishing test h_i . Forger uses boosting algorithm to produce u_{i+1}.

- In few iterations, either critic fails (indistinguishability) or boosting algorithm produces strong hypothesis = counter-example for d-pseudo-density

Consequence

- Using any appropriate boosting algorithm, and any distinguisher, we obtain an efficient method for Generation by Adversarial Boosting (GAB) , which produces either an indistinguishable distribution or a counter-example to pseudo-density d.

- Best number of iterations: O (log 1/d / e^2). (However, finding samples from u takes time O(1/d)).

GAB algorithm: the distinguisher

- DIST is a distinguisher. It takes as input two distributions, S and D, and tries to find an h in H where h(S) > h(D) + e. It could be stateful, if desired, or stateless. It could be a neural net classifier, but it doesn't have to be. Sometimes, we will be able to show a particular algorithm Dist that is complete for H: if there is such a distinguishing test h, Dist will find one that comes close.

- We think of Dist as the critic.

Form of distribution

- Say in the past, Dist has produced h_1,..h_i
- Define s_i(x) = 2(h_1(x) + h_2(x)…+h_i(x)) -i, number ``S-like'' minus number ``non-S-like''
- v is an integer valued variable, decreasing over time
- M[s] = min (1, exp (es))
- u_{i,v}(x) = M(s_i (x) - v) is a measure
- D_{u_{i,v}} is the induced distribution

GAB

- T:= C log (1/d)/e^2.
- v := ln 1/d, i:=0 (So u_{0,v} (x) =d.)
- Repeat T times or until Dist fails:
    - If Dist fails on S and D_{u_{i,v}}, return u_{i,v}
    - ELSE h_{i+1} is the hypothesis produced by Dist
    - IF d (u_{i+1,v}) < d THEN decrement v
- Find a t so that Threshold_t (h_1(x)+…h_T(x)) is a counter-example to (d,e)-pseudo-density for S

Theoretical guarantee

- GAB either produces a distribution D_{u_{i,v}} of density at least d which Dist fails to distinguish from S, or a counter-example to pseudo-density.

Win-win

- If Dist is a good distinguisher, fooling Dist means that the distribution produced ``looks like S''. Density d means the distribution is ``very random''

- Finding a counter-example to pseudo-density means characterizing the range of S, showing that it is ``non-random''

Criticism of GAN

- Arora, Ge, Liang, Ma, Zhang: GANs might greatly reduce entropy of distribution, approaching ``replays''

- Arora, Zhang: Empirical study shows this happening with real applications of GANs

Why entropy loss is a problem

Generation by Adversarial Reflective Boosting (GARB)

- Modification of GAB to include looking for symmetric function of current hypotheses as distinguishers, i.e., using measure as distinguisher

- Cost: Distributions, counter-examples are deep symmetric circuits of hypotheses

- Advantage: If S is pseudo-dense, we find a model that is both indistinguishable and has at least the entropy of S (up to small additive loss, o(1) bits.).

GARB

- We keep GAB as is. The only change is in the Distinguisher.

- ReflectiveDist (S, u_{i,v} = M(s_i -v))

- Use samples from both distributions to make histograms of s_i

- If the distributions are greater than e statistical distance then there is a symmetric function h_{i+1} of h_1…h_i that distinguishes S and D_{u_{i,v}}. Return h_{i+1}

- Else, return Dist (S, D_{u_{i,v}}).

Properties of GARB

- h_k is a depth at most k symmetric circuit in functions from H.

- Same guarantees as GAB, because special case.

- If Dist' fails, Entropy (D_{u_{i,v}}) > Entropy (S) - O(e log (1/ed)), so almost maximal entropy.

- The same is true for any S' that is indistinguishable from S via symmetric circuits over H, so if S is sub-sampled from S', the guarantee will hold for the entropy of S'

What can go wrong?

- Distribution not pseudo-dense for reasonable d. How likely are random matrices of pixels to look like cats? Then we find characterization, can apply recursively, if we can sample within subset defined by characterization. Or start with a smaller space to begin with, by identifying useful features before algorithm starts. (Pictures of cats within space of pictures of objects, not within random noise.)

- Weak distinguisher. Then we'll find a distribution that is high-entropy, but stronger distinguishers might tell the difference. Fix one: many situations, provable distinguishers (e.g., cut norm for graphs, low dimensional geometric distributions). Fix two: resume GARB with stronger distinguisher if unsatisfied.

Mathematical consequences

- In addition to possible machine learning applications, we can apply machine learning tools to mathematics using the dense model theorem with better boosting algorithms

- Graph theory: Improved weak regularity lemma for sparse graphs

- Fourier analysis: Improve Chang's Inequality for Boolean functions that don't have huge intersections with sub-spaces

- Computational complexity: Characterization of pseudo-entropy

- Cryptography: Leakage-resistant pseudo-random strings.

Leakage resilient cryptography

- In crypto, randomness is cheap, but secret randomness is expensive.

- Say we have a (shared) secret random string R, but the adversary has managed to learn some k -bits of information about R, L(R)

- Can use a (strong) randomness Extractor with a public randomly chosen seed to extract secret R'=Ext(s, R) that will be a slightly smaller secret random string

Pseudo-random secret

- But usually R isn't random, but pseudo-random, R=G(r). In that case, the leaked bits could be functions of r. Do extractors still work?

- Yes: Dense Model theorem: Distribution (R, L(r)) is 2^{-k} pseudo-dense, has a dense model. Extracting from this model is random, given L, so extracting from R is indistinguishable from random given L. GAB: also true in uniform model

Applied to Regularity Lemmas

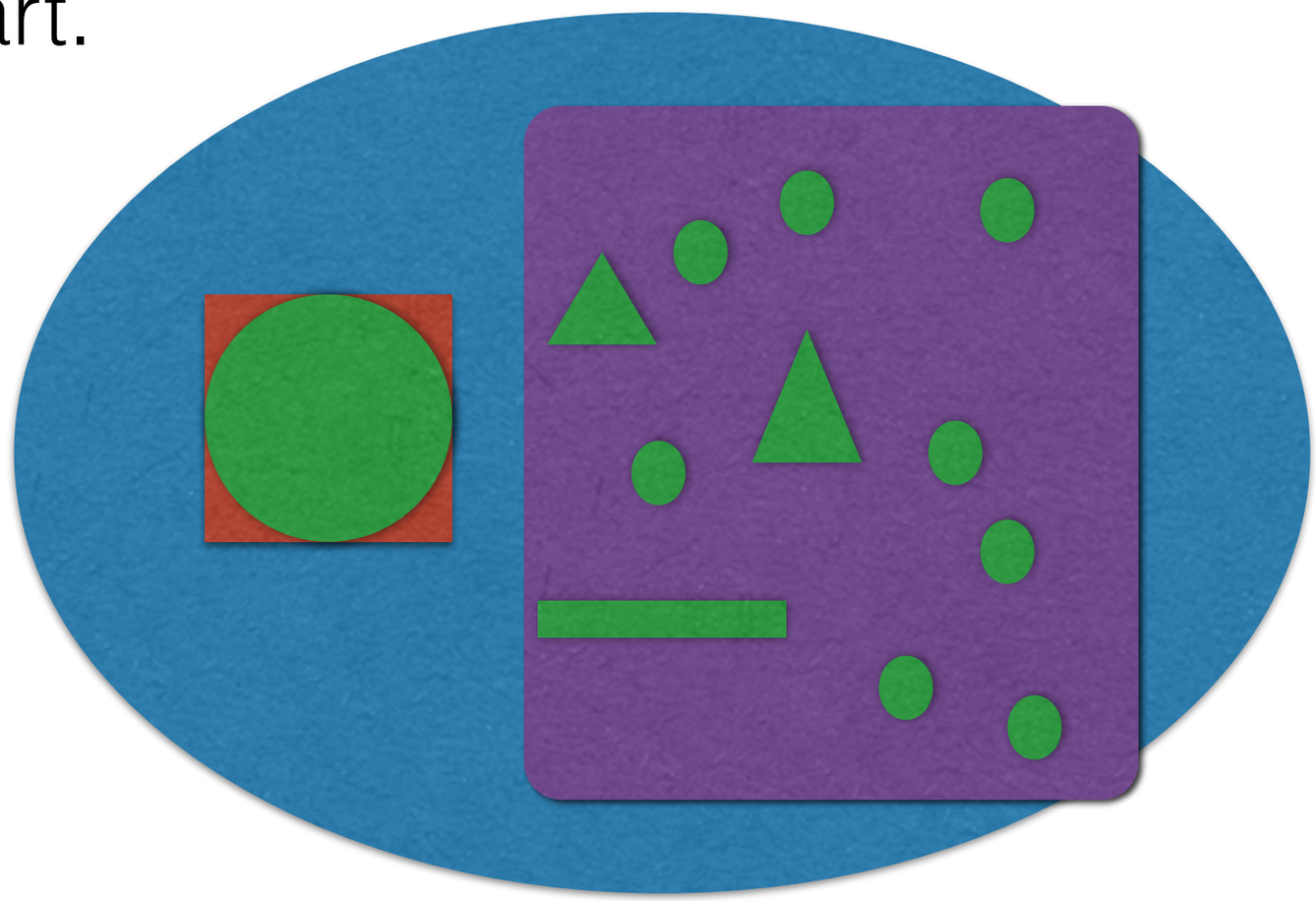
- Works for non-complete graphs, e.g., multi-partite graphs, conditioning on degrees, etc.

- Dense Model theorem gives weak regularity lemma for sparse graphs: 2^O(log 1/d e^{-2})) rather than 2^O(d^{-2} e^{-2})

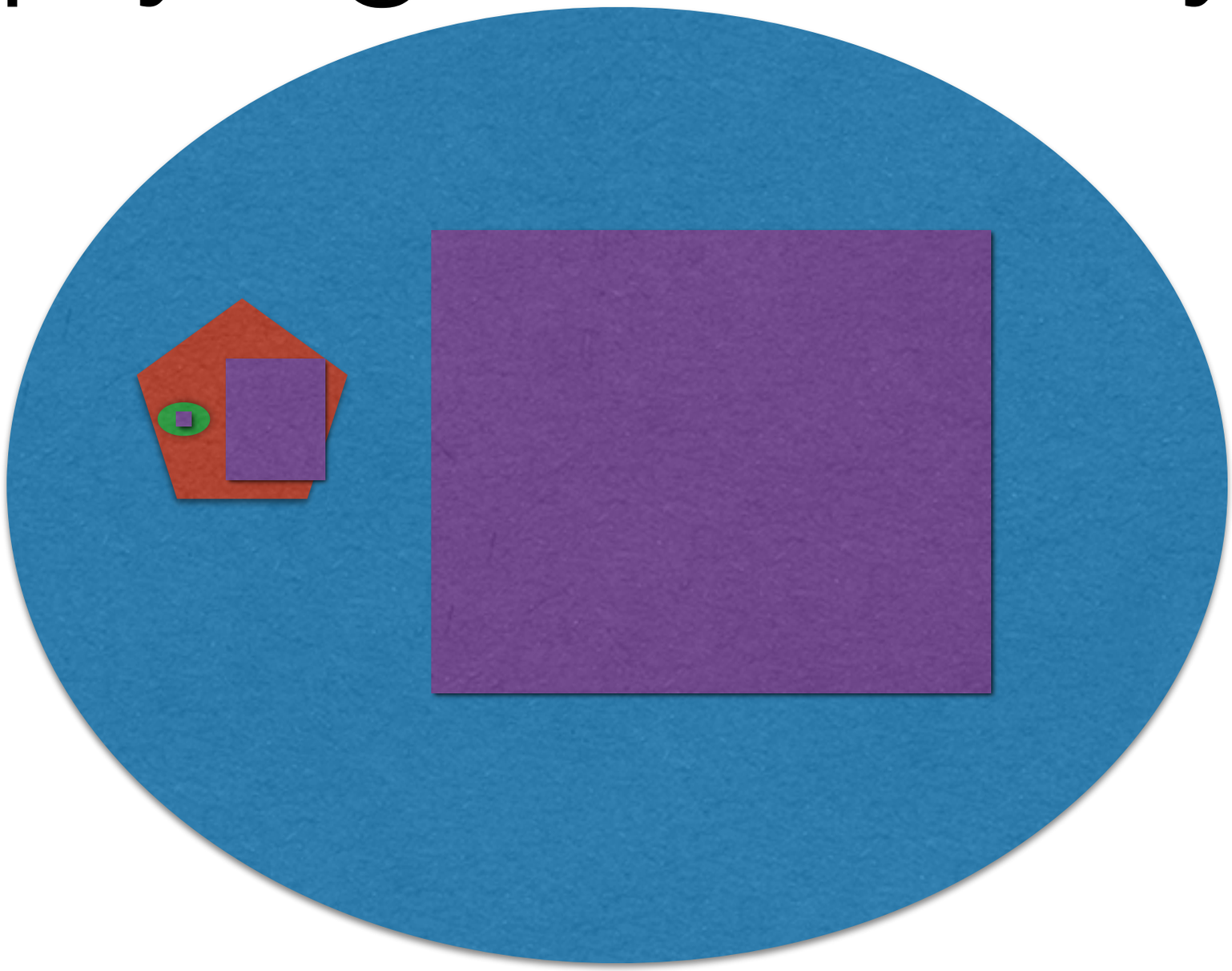
- While generic result only for Weak Regularity, can reverse engineer to produce ``boost 'til you bust'', a boosting version that implies Szemeredi Regularity

Decomposition

- Instead of an ``either-or'', we can use DML iteratively to decompose an arbitrary distribution into a part that is non-dense and a model of the dense part.



Applying recursively



- Telescoping model with k =poly(1/e , log 1/d)

Pseudo-entropy

- Applying this in turn to computational complexity, this shows that if a distribution has pseudo-entropy k , then it has a model D’ that is indistinguishable, has entropy k - e, and where D’ is samplable in P^NP / poly

Conclusion

- This connection allows us to draw on powerful machine learning tools for a wide variety of mathematical subjects

- ``Plug and play'' theorems allow you to try to find the boosting algorithm that gives you the best regularity lemmas for your application

- We can also hope to use the connection to prove theorems explaining the success of machine learning techniques, and extending those techniques.