Regularization Methods for Sequence Analysis

Gunnar Rätsch

Friedrich Miescher Laboratory, Max Planck Society
Tübingen, Germany

Workshop "Regularisierte Schätzung zur Modellierung molekularer Daten"
GMDS Jahrestagung in Mannheim

September 6, 2010

Group's Research Topics

Machine Learning

1. Statistical inference methods for structured data
   $\Rightarrow$ *Develop fast and accurate learning methods*
2. Convergence properties of iterative algorithms
   $\Rightarrow$ *Boosting-like algorithms and semi-infinite LPs*

Molecular Biology

3. Genome annotation
   $\Rightarrow$ *Predict features encoded on DNA*
4. Biological networks
   $\Rightarrow$ *Understand interactions between gene products*
5. Analysis of polymorphisms
   $\Rightarrow$ *Discover polymorphisms and associate with phenotypes*

Group's Research Topics

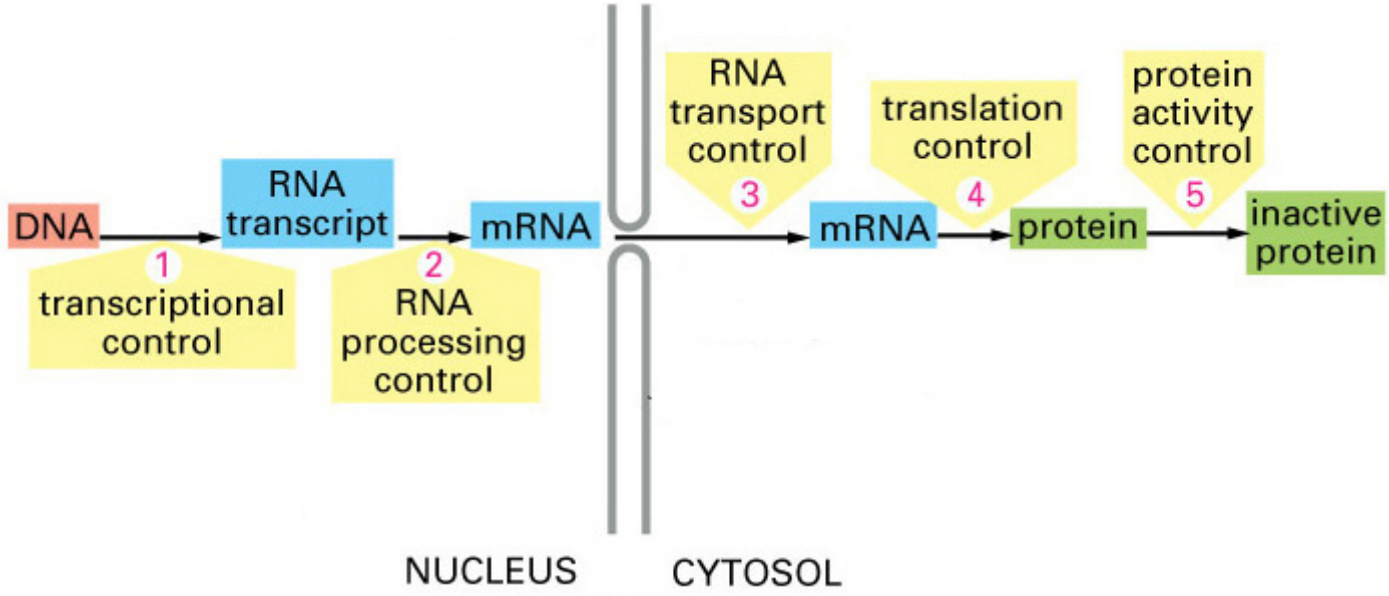
Machine Learning

1. Statistical inference methods for structured data
   ⇒ *Develop fast and accurate learning methods*
2. Convergence properties of iterative algorithms
   ⇒ *Boosting-like algorithms and semi-infinite LPs*

Molecular Biology

3. Genome annotation
   ⇒ *Predict features encoded on DNA*
4. Biological networks
   ⇒ *Understand interactions between gene products*
5. Analysis of polymorphisms
   ⇒ *Discover polymorphisms and associate with phenotypes*

Learning about the Transcriptome



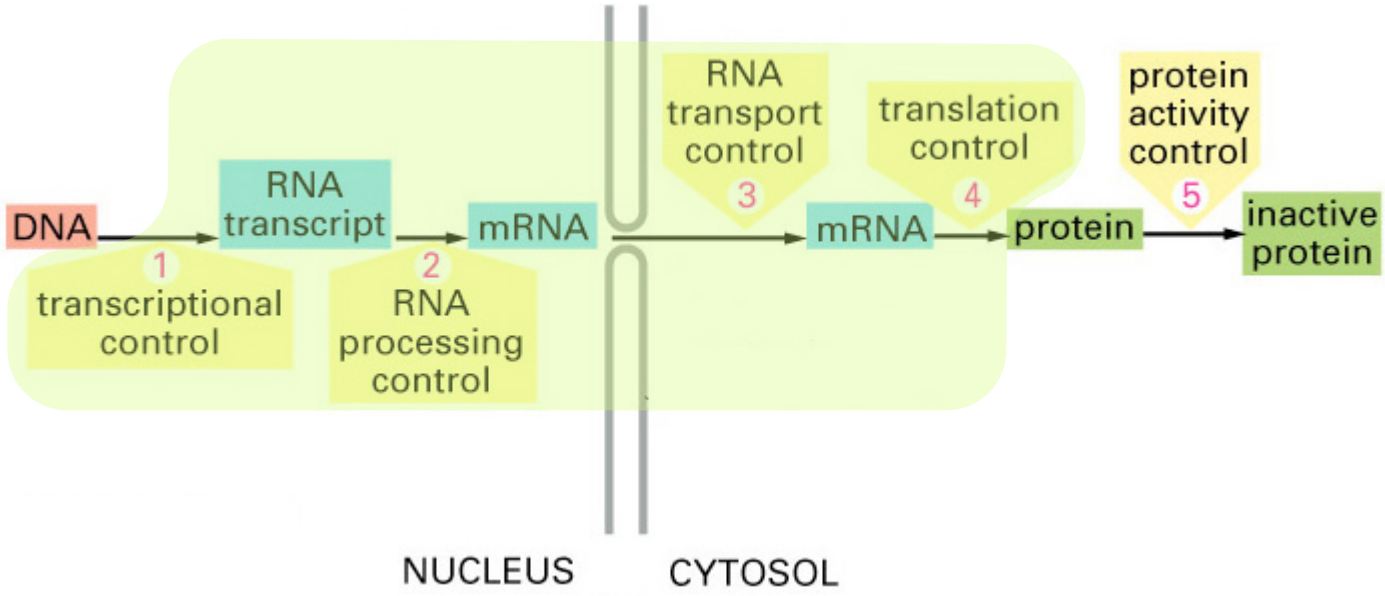
Machine Learning View

- How to learn to predict what these processes accomplish?
- How well can we predict it from the available information?

Biological View

1. What can we not predict yet? What is missing?
2. Can we derive a deeper understanding of these processes?

Learning about the Transcriptome



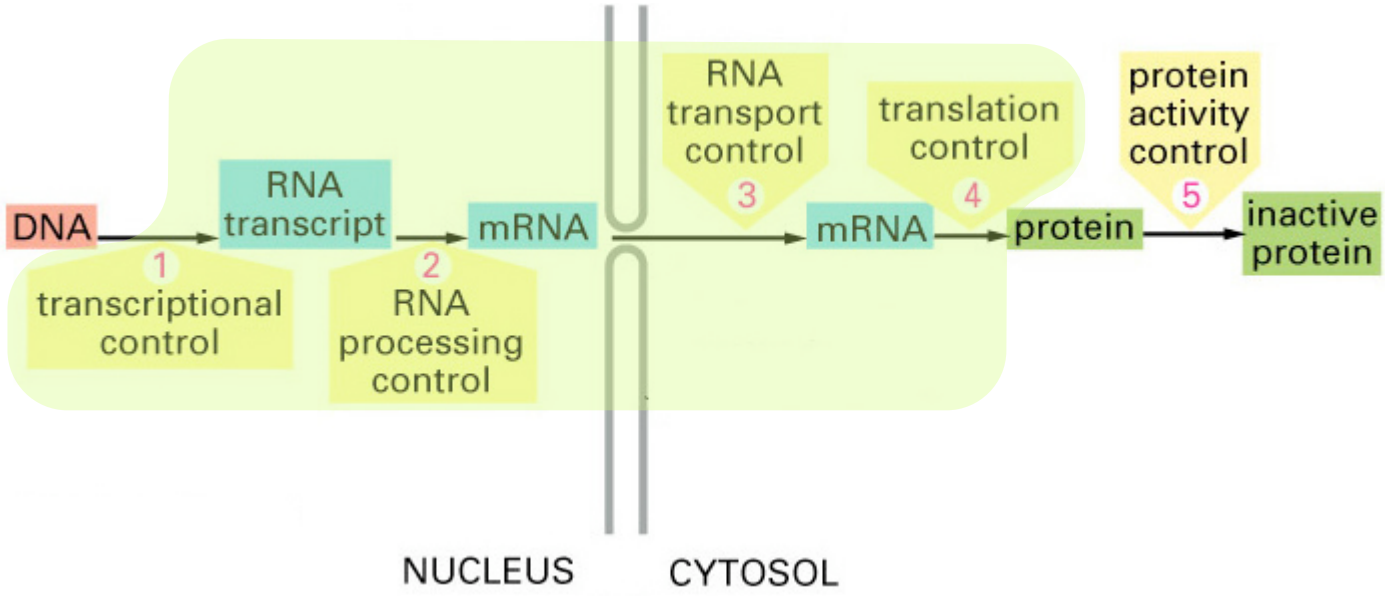
Machine Learning View

- How to learn to predict what these processes accomplish?
- How well can we predict it from the available information?

Biological View

1. What can we not predict yet? What is missing?
2. Can we derive a deeper understanding of these processes?

Learning about the Transcriptome



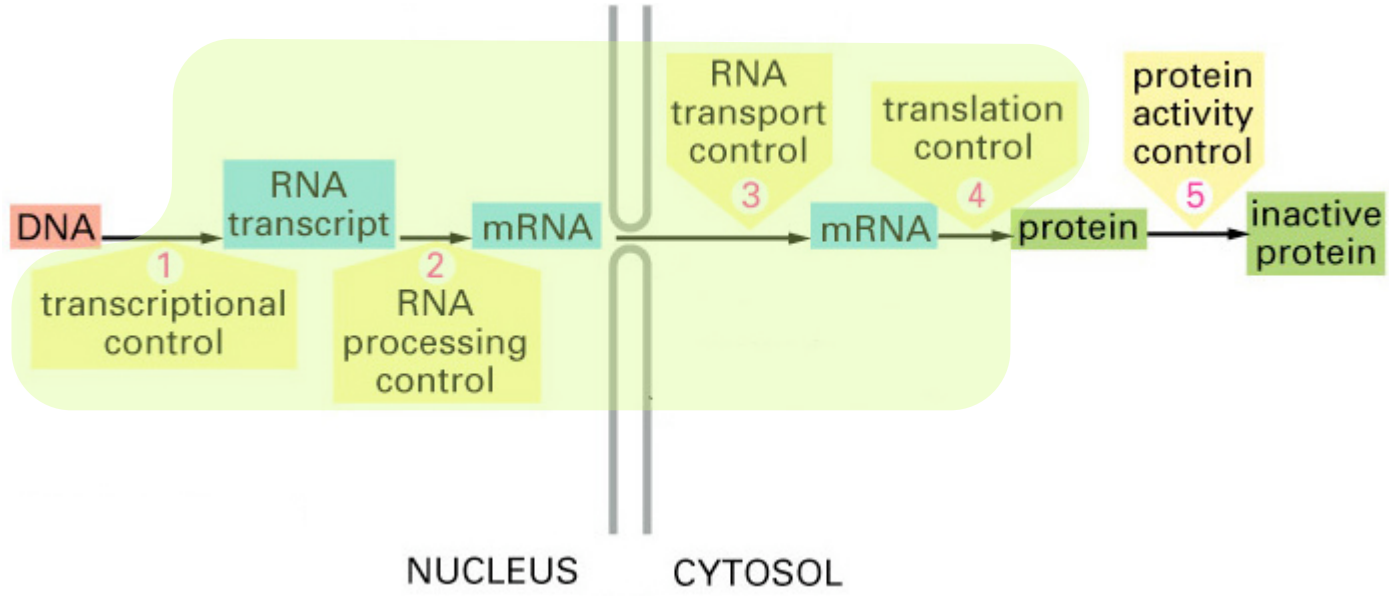
Machine Learning View

- How to learn to predict what these processes accomplish?
- How well can we predict it from the available information?

Biological View

1. What can we not predict yet? What is missing?
2. Can we derive a deeper understanding of these processes?

Learning about the Transcriptome



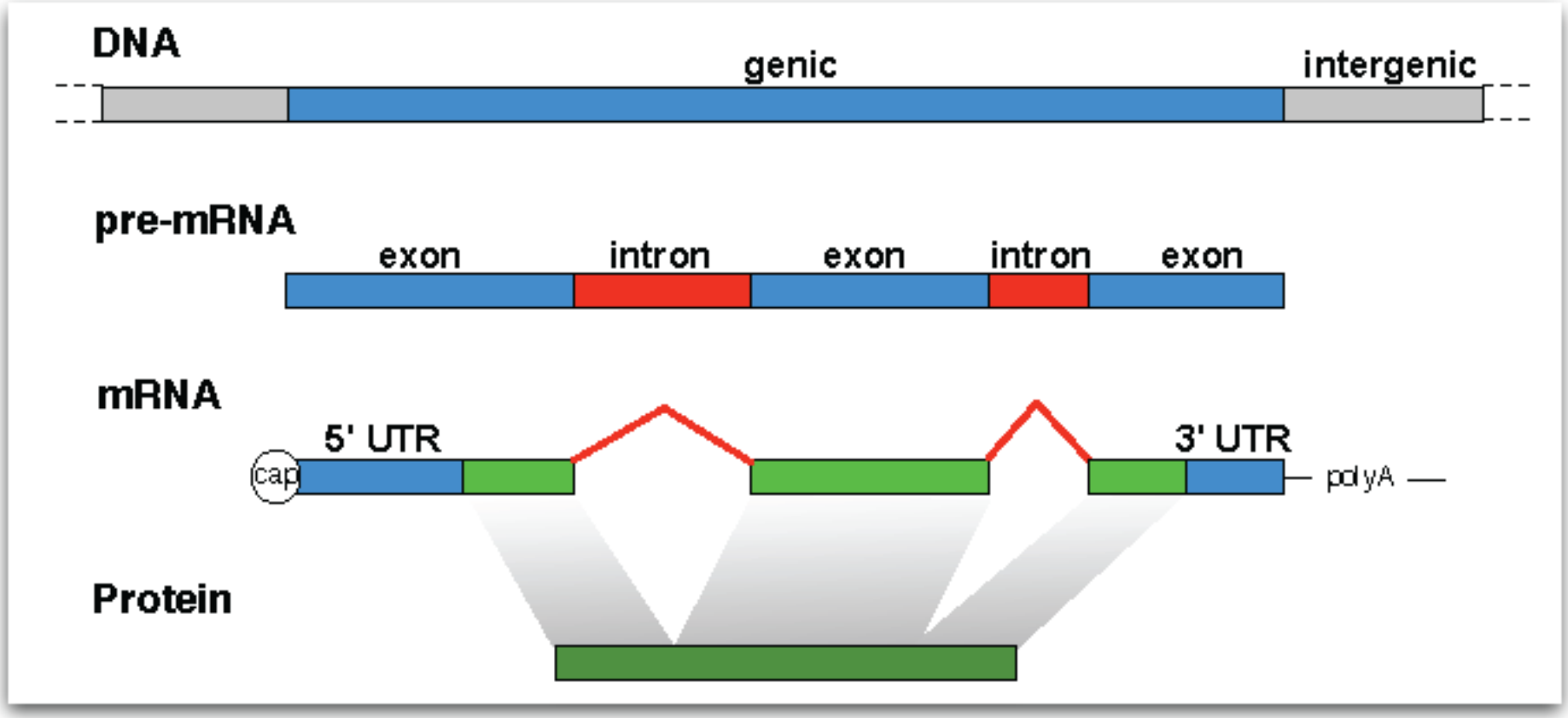
Machine Learning View

- How to learn to predict what these processes accomplish?
- How well can we predict it from the available information?

Biological View

1. What can we not predict yet? What is missing?
2. Can we derive a deeper understanding of these processes?

Computational Genome Annotation



Simplest formulation:

- Given a DNA sequence $\mathbf{x} \in \{\text{'A','C','G','T'}\}^L$
- Find the correct **label sequence y** $= y_1 y_2 \ldots y_L$
  ($y_i \in Y = \{$"intergenic", "5' UTR", "coding", "intron", $\ldots\}$)

Discriminative Segmentation Learning

- Traditionally HMMs, here: Max-Margin Structured Output Learning
- Learn function $f(\mathbf{y}|\mathbf{x})$ scoring segmentations $\mathbf{y}$ for $\mathbf{x}$
- Maximize $f(\mathbf{y}|\mathbf{x})$ w.r.t. $\mathbf{y}$ for prediction:

$$\operatorname*{argmax}_{\mathbf{y}\in\Upsilon^*} f(\mathbf{y}|\mathbf{x})$$

- Idea: $f(\mathbf{y}|\mathbf{x}) \gg f(\hat{\mathbf{y}}|\mathbf{x})$ for wrong labels $\hat{\mathbf{y}} \neq \mathbf{y}$
- Max-margin learning: [Altun et al., 2003, Rätsch and Sonnenburg, 2007]
  - Given N sequence pairs $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)$ for training
  - Solve optimization problem:

$$\max_{\rho, f \in \mathcal{H}} \quad \rho - C \sum_{n=1}^{N} \xi_n$$

$$\text{w.r.t.} \quad f(\mathbf{y}_n|\mathbf{x}_n) - f(\mathbf{y}|\mathbf{x}_n) \geq \rho \quad \xi_n$$

$$\text{for all } \mathbf{y} \neq \mathbf{y}_n \in \Upsilon^*, n = 1, \ldots, N$$

Discriminative Segmentation Learning

- Traditionally HMMs, here: Max-Margin Structured Output Learning
- Learn function $f(\mathbf{y}|\mathbf{x})$ scoring segmentations $\mathbf{y}$ for $\mathbf{x}$
- Maximize $f(\mathbf{y}|\mathbf{x})$ w.r.t. $\mathbf{y}$ for prediction:

$$\operatorname*{argmax}_{\mathbf{y}\in\Upsilon^*} f(\mathbf{y}|\mathbf{x})$$

- Idea: $f(\mathbf{y}|\mathbf{x}) \gg f(\hat{\mathbf{y}}|\mathbf{x})$ for wrong labels $\hat{\mathbf{y}} \neq \mathbf{y}$
- Max-margin learning: [Altun et al., 2003, Rätsch and Sonnenburg, 2007]
  - Given N sequence pairs $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)$ for training
  - Solve optimization problem:

$$\max_{\rho, f, \xi \in \mathbb{R}^N_+} \quad \rho - C\sum_{n=1}^{N} \xi_n$$

$$\text{w.r.t.} \quad f(\mathbf{y}_n|\mathbf{x}_n) - f(\mathbf{y}|\mathbf{x}_n) \geq \rho - \xi_n$$

$$\text{for all } \mathbf{y} \neq \mathbf{y}_n \in \Upsilon^*, n = 1, \ldots, N$$

Discriminative Segmentation Learning

- Traditionally HMMs, here: Max-Margin Structured Output Learning
- Learn function $f(\mathbf{y}|\mathbf{x})$ scoring segmentations $\mathbf{y}$ for $\mathbf{x}$
- Maximize $f(\mathbf{y}|\mathbf{x})$ w.r.t. $\mathbf{y}$ for prediction:

$$\operatorname*{argmax}_{\mathbf{y}\in\Upsilon^*} f(\mathbf{y}|\mathbf{x})$$

- Idea: $f(\mathbf{y}|\mathbf{x}) \gg f(\hat{\mathbf{y}}|\mathbf{x})$ for wrong labels $\hat{\mathbf{y}} \neq \mathbf{y}$
- Max-margin learning: [Altun et al., 2003, Rätsch and Sonnenburg, 2007]
  - Given N sequence pairs $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)$ for training
  - Solve optimization problem:

$$\begin{aligned} \max_{\rho, f, \boldsymbol{\xi}\in\mathbb{R}^N_+} \quad & \rho - C\sum_{n=1}^{N}\xi_n \\ \text{w.r.t.} \quad & f(\mathbf{y}_n|\mathbf{x}_n) - f(\mathbf{y}|\mathbf{x}_n) \geq \rho - \xi_n \\ & \text{for all } \mathbf{y} \neq \mathbf{y}_n \in \Upsilon^*, n = 1, \ldots, N \end{aligned}$$

Discriminative Segmentation Learning

- Traditionally HMMs, here: Max-Margin Structured Output Learning
- Learn function $f(\mathbf{y}|\mathbf{x})$ scoring segmentations $\mathbf{y}$ for $\mathbf{x}$
- Maximize $f(\mathbf{y}|\mathbf{x})$ w.r.t. $\mathbf{y}$ for prediction:

$$\operatorname*{argmax}_{\mathbf{y}\in\Upsilon^*} f(\mathbf{y}|\mathbf{x})$$

- Idea: $f(\mathbf{y}|\mathbf{x}) \gg f(\hat{\mathbf{y}}|\mathbf{x})$ for wrong labels $\hat{\mathbf{y}} \neq \mathbf{y}$
- Max-margin learning: [Altun et al., 2003, Rätsch and Sonnenburg, 2007]
  - Given N sequence pairs $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)$ for training
  - Solve optimization problem:

$$\begin{aligned} \max_{\rho, f, \boldsymbol{\xi}\in\mathbb{R}^N_+} \quad & \rho - C\sum_{n=1}^{N}\xi_n \\ \text{w.r.t.} \quad & f(\mathbf{y}_n|\mathbf{x}_n) - f(\mathbf{y}|\mathbf{x}_n) \geq \rho - \xi_n \\ & \text{for all } \mathbf{y} \neq \mathbf{y}_n \in \Upsilon^*, n = 1, \ldots, N \end{aligned}$$

Optimization Strategy ($f(\mathbf{y}|\mathbf{x}) = \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle$)

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\xi}\geq 0} \quad & \|\mathbf{w}\|_? + C \sum_{n=1}^{N} \xi_n \\ \text{w.r.t.} \quad & \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}_n) - \Phi(\mathbf{x}, \mathbf{y}) \rangle \geq 1 - \xi_n \\ & \text{for all } \mathbf{y}_n \neq \mathbf{y} \in \Upsilon^*, n = 1, \ldots, N \end{aligned}$$

- Big: one constraint per example and wrong labeling
- Iterative solution (a.k.a. "column generation", "cutting planes")
  - Begin with small set of wrong labellings
  - Solve reduced optimization problem
  - Find labellings that violate constraints
  - Add constraints, resolve
- Iteration bounds:
  - $\|\cdot\|_2$: R/ϵ^2, see [Tsochantaridis et al., 2005]
  - $\|\cdot\|_1$: $\log(N)/\epsilon^2$ [Rätsch and Warmuth, 2005], $\mathcal{O}(1/\epsilon)$ [Warmuth et al., 2009]

Problems

- Optimization may require many iterations
- Number of constraints increases linearly
- When using kernels, solving optimization problems can become easily infeasible (number of variables also increases)

$$\mathbf{w} = \sum_{n=1}^{N} \sum_{\mathbf{y} \in \Upsilon^*} \alpha_{n,\mathbf{y}} \Phi(\mathbf{x}_n, \mathbf{y})$$

- Approximation algorithms needed (or better optimization alg.)
- Idea: Decompose problem
  - First part uses kernels, can be precomputed
  - Second part without kernels and only combines ingredients

$\Rightarrow$ Solve problems with $10,000$ examples, instead of just ≤ 500

Problems

- Optimization may require many iterations
- Number of constraints increases linearly
- When using kernels, solving optimization problems can become easily infeasible (number of variables also increases)

$$\mathbf{w} = \sum_{n=1}^{N} \sum_{\mathbf{y} \in \Upsilon^*} \alpha_{n,\mathbf{y}} \Phi(\mathbf{x}_n, \mathbf{y})$$

- Approximation algorithms needed (or better optimization alg.)
- Idea: Decompose problem
  - First part uses kernels, can be precomputed
  - Second part without kernels and only combines ingredients

$\Rightarrow$ Solve problems with $10,000$ examples, instead of just ≤ 500

Problems

- Optimization may require many iterations
- Number of constraints increases linearly
- When using kernels, solving optimization problems can become easily infeasible (number of variables also increases)

$$\mathbf{w} = \sum_{n=1}^{N} \sum_{\mathbf{y} \in \Upsilon^*} \alpha_{n,\mathbf{y}} \Phi(\mathbf{x}_n, \mathbf{y})$$

- Approximation algorithms needed (or better optimization alg.)
- Idea: Decompose problem
  - First part uses kernels, can be precomputed
  - Second part without kernels and only combines ingredients

$\Rightarrow$ Solve problems with $10,000$ examples, instead of just ≤ 500

Problems

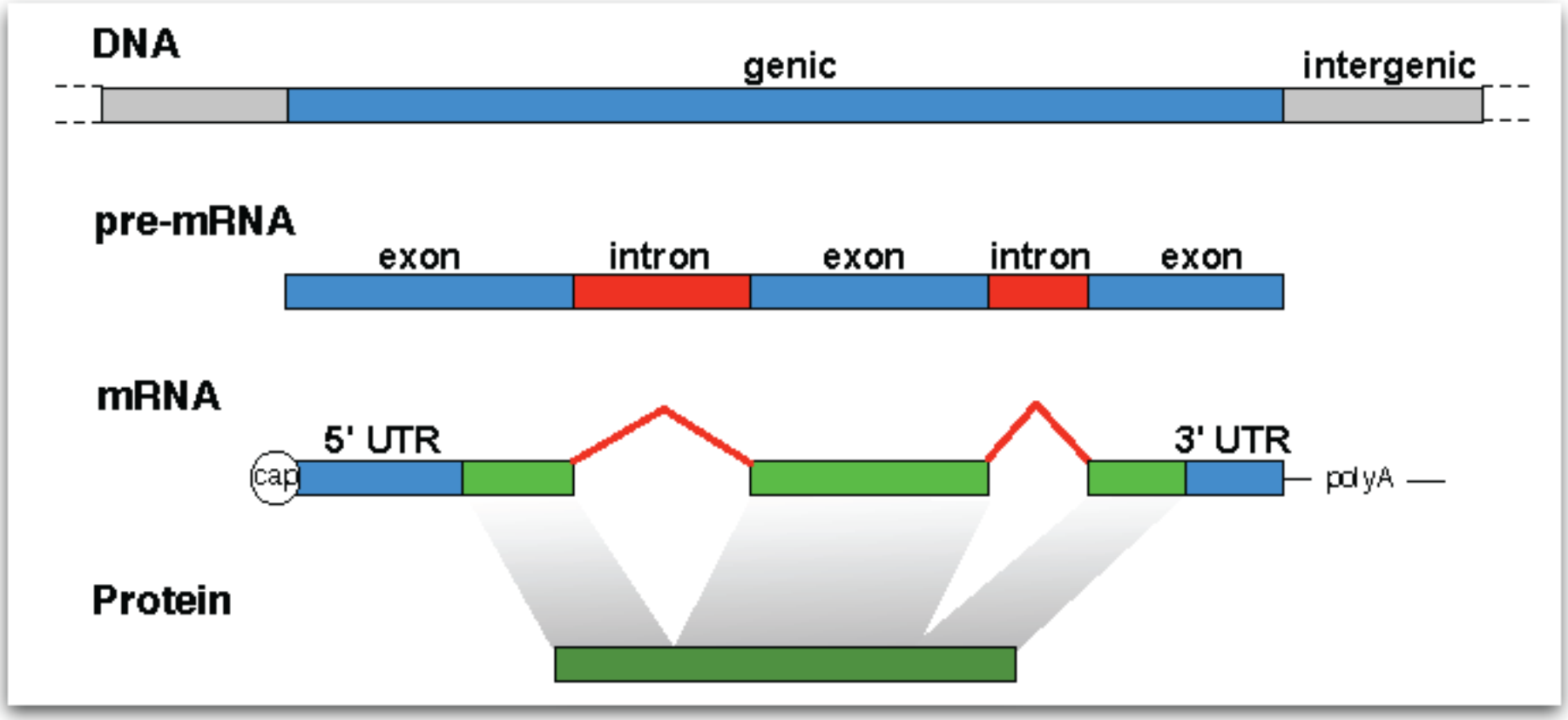
- Optimization may require many iterations
- Number of constraints increases linearly
- When using kernels, solving optimization problems can become easily infeasible (number of variables also increases)

$$\mathbf{w} = \sum_{n=1}^{N} \sum_{\mathbf{y} \in \Upsilon^*} \alpha_{n,\mathbf{y}} \Phi(\mathbf{x}_n, \mathbf{y})$$

- Approximation algorithms needed (or better optimization alg.)
- Idea: Decompose problem
  - First part uses kernels, can be precomputed
  - Second part without kernels and only combines ingredients

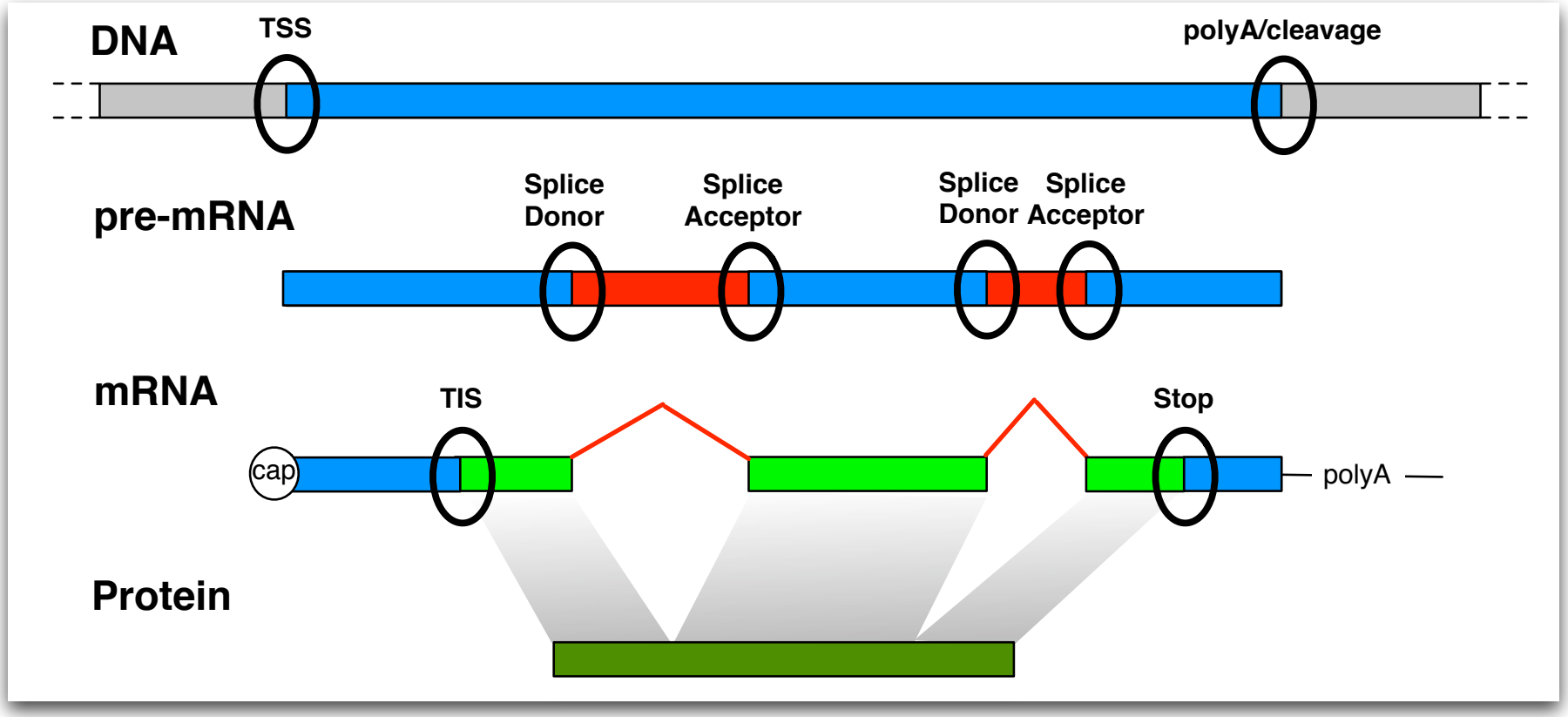
$\Rightarrow$ Solve problems with $10,000$ examples, instead of just ≤ 500

Back to Computational Gene Finding



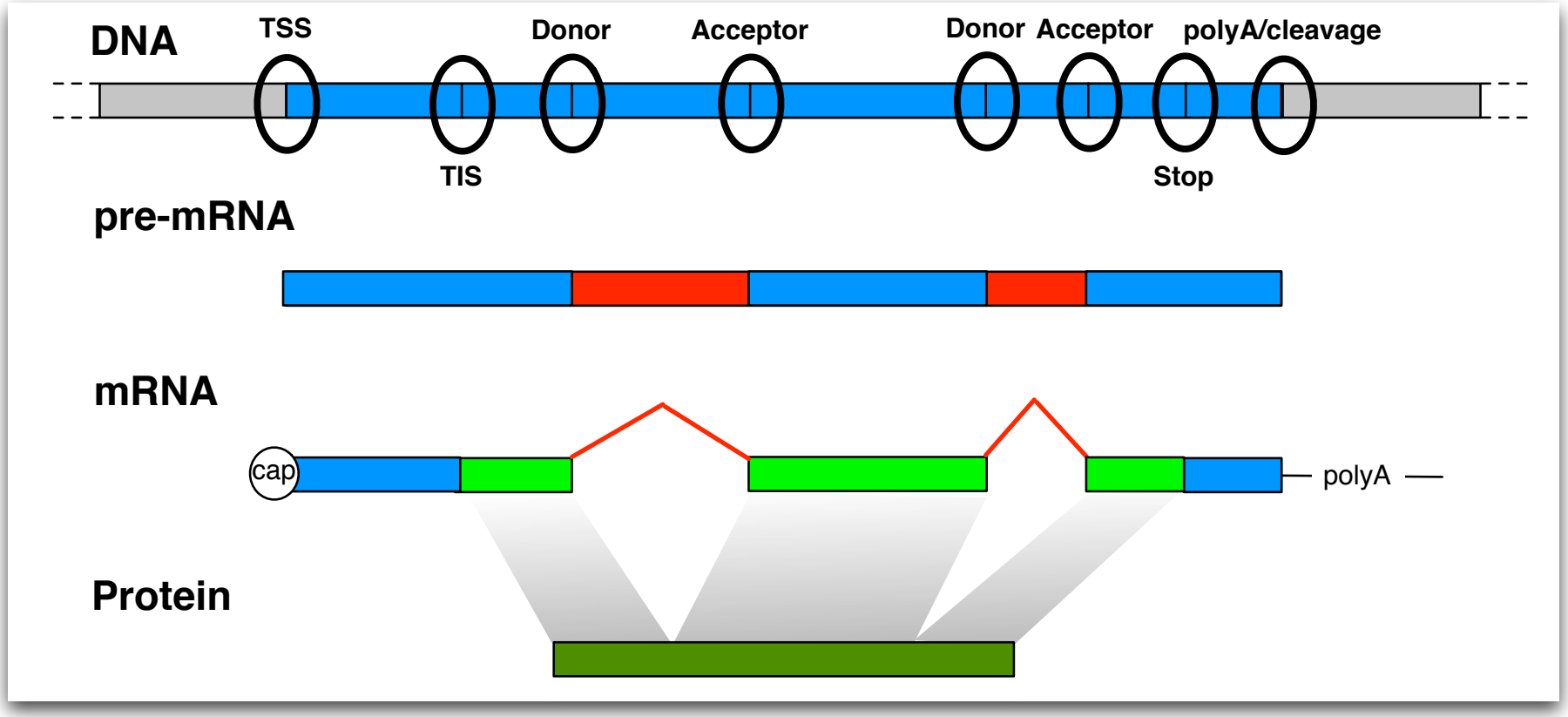
- Given a piece of DNA sequence
- Predict gene products inluding the intermediate processing steps

Back to Computational Gene Finding



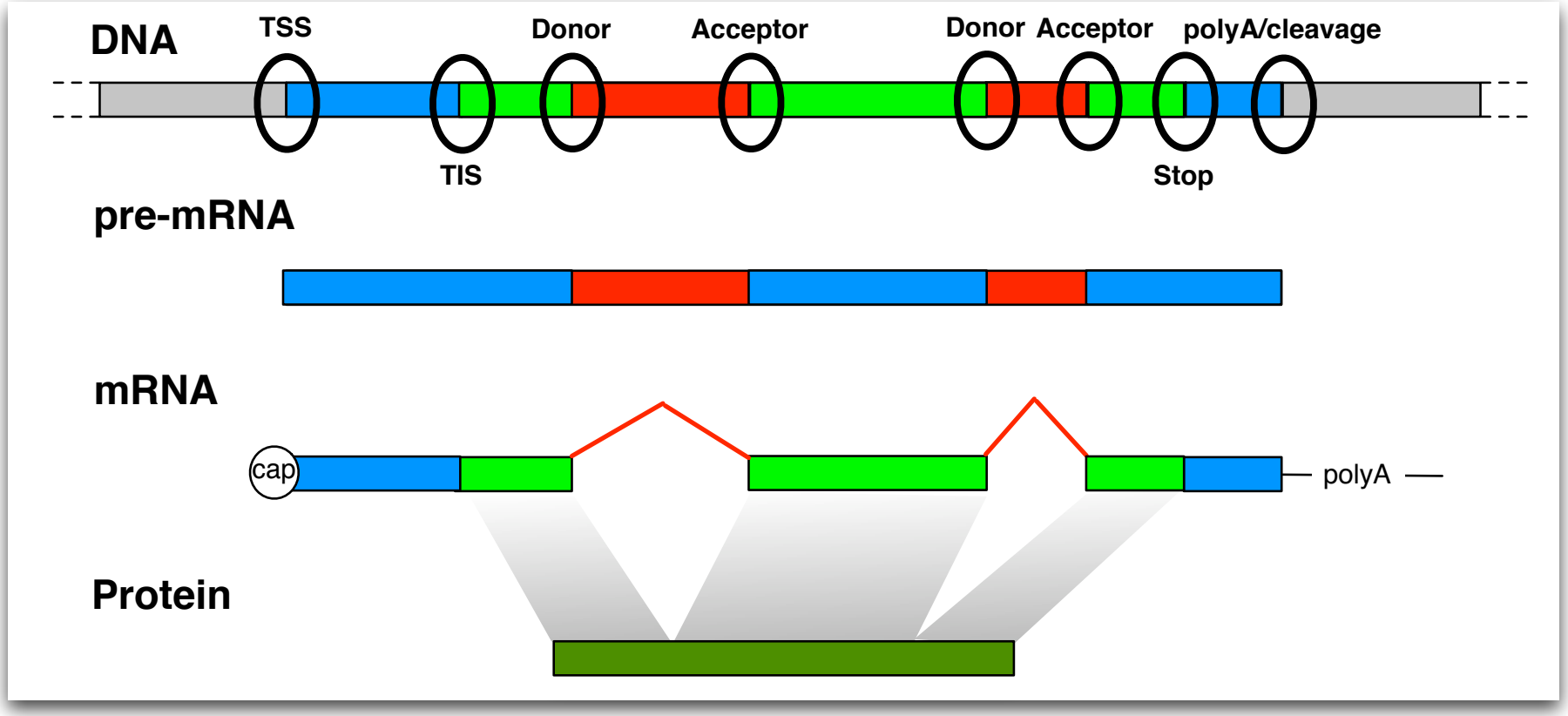
- Predict signals used during processing

Back to Computational Gene Finding



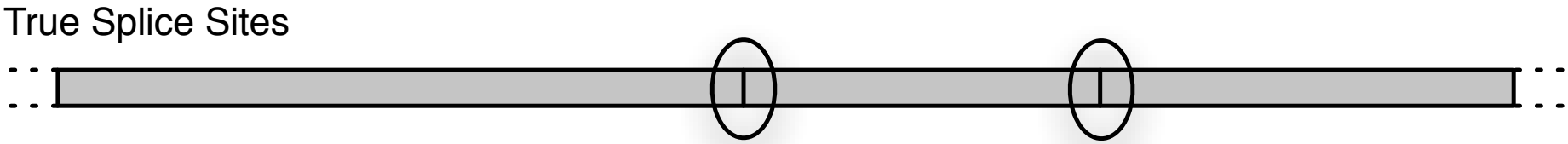
- Predict signals used during processing

Back to Computational Gene Finding

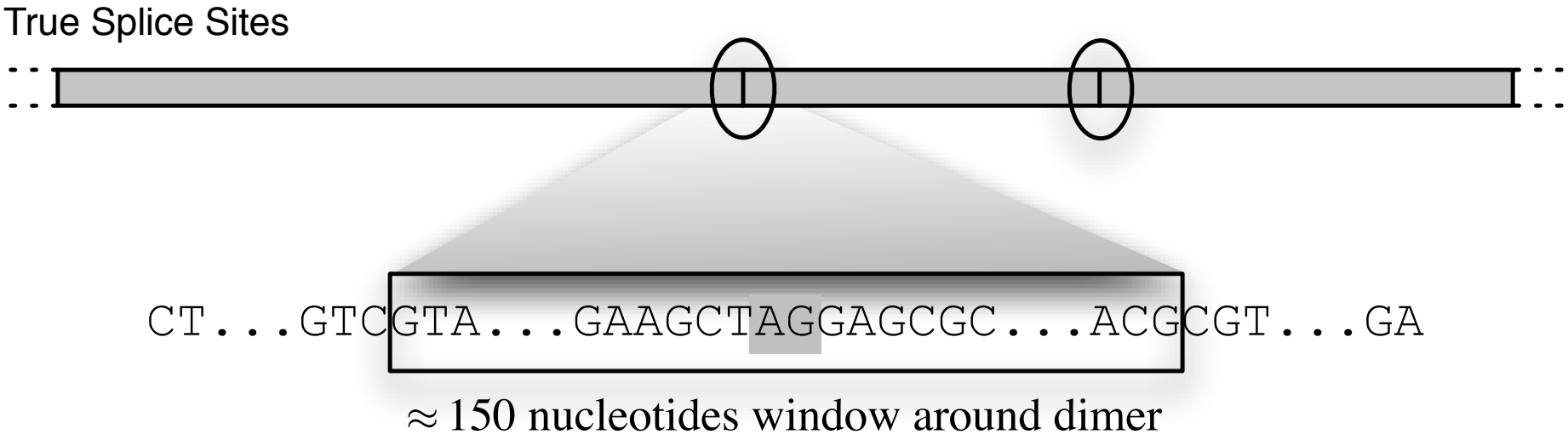


- Predict signals used during processing
- Predict the correct corresponding **label sequence** with labels "intergenic", "exon", "intron", "5' UTR", etc.

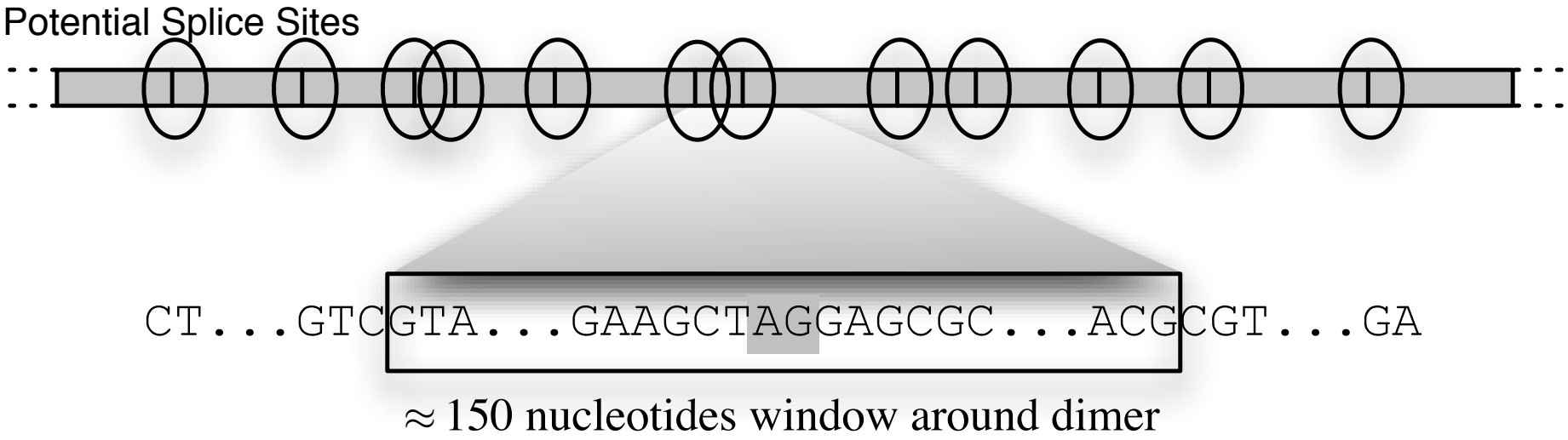
Example: Splice Site Recognition



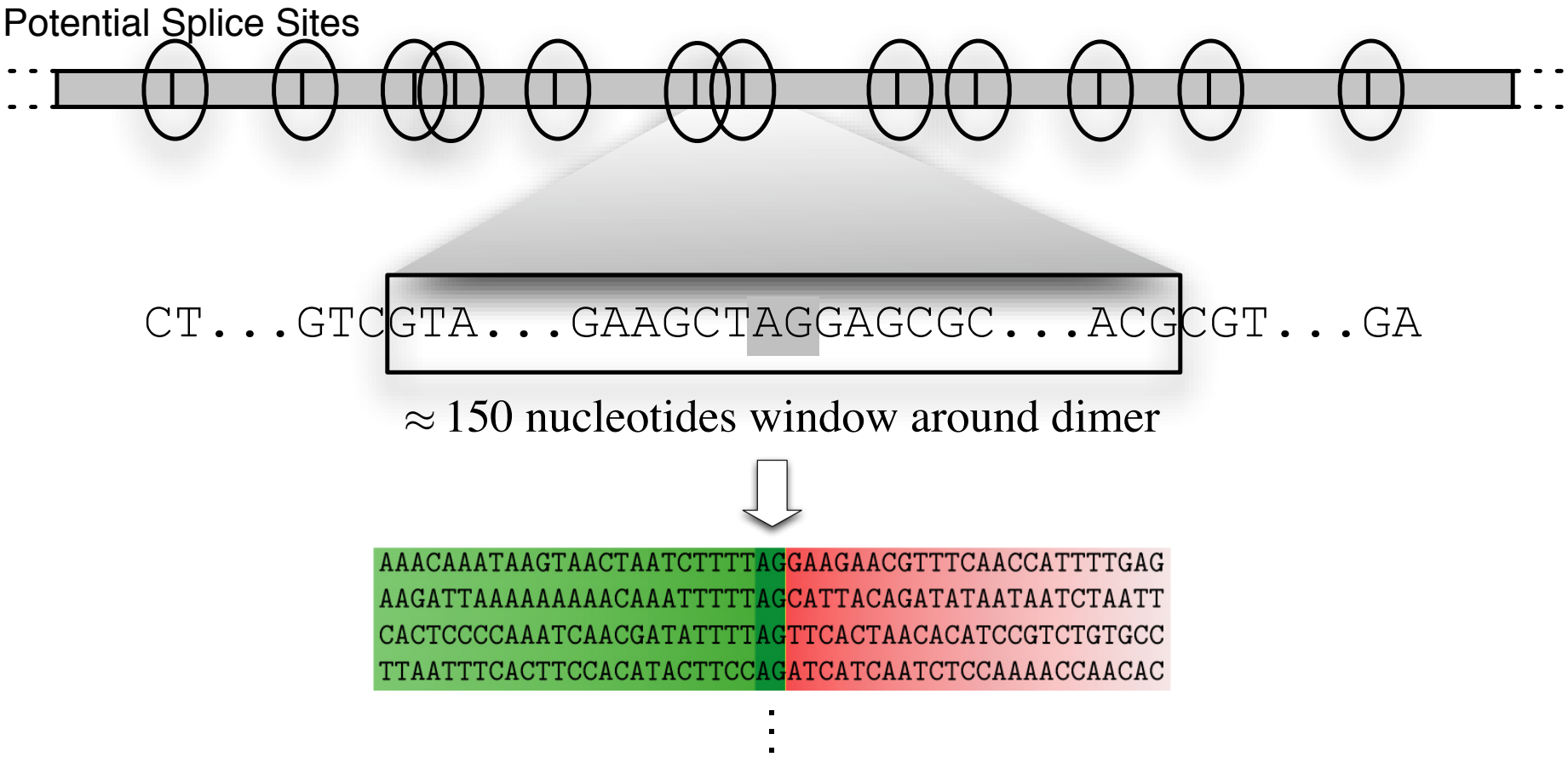
Example: Splice Site Recognition



Example: Splice Site Recognition



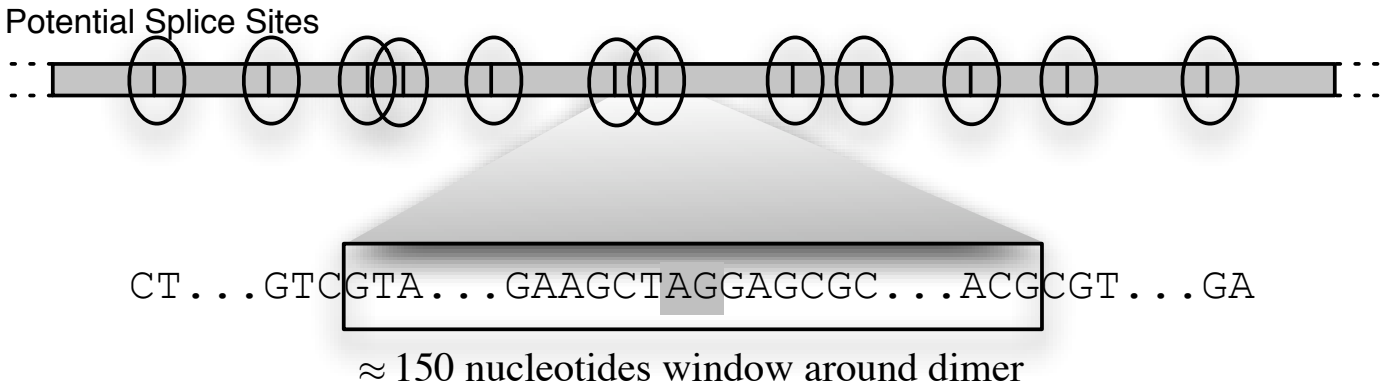
Example: Splice Site Recognition



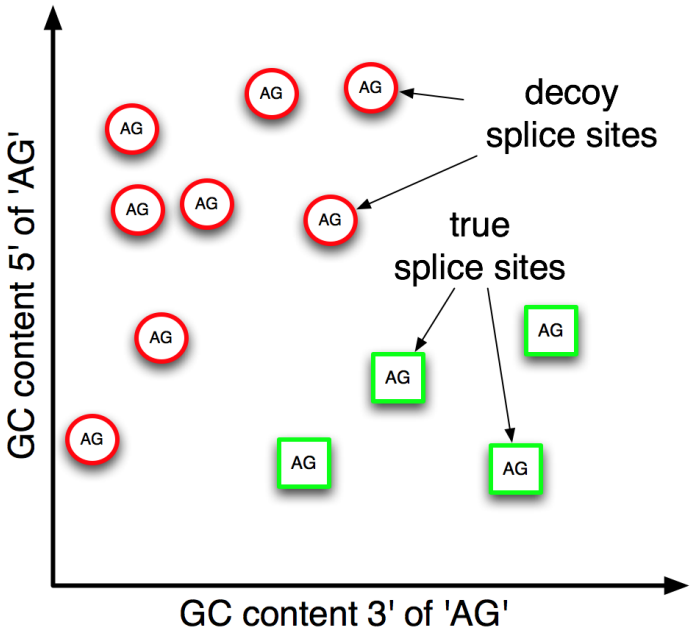
- True sites: fixed window around a true splice site
- Decoy sites: all other consensus sites

$\Rightarrow$ **Millions of examples from EST databases**

Example: Splice Site Recognition



Basic idea:



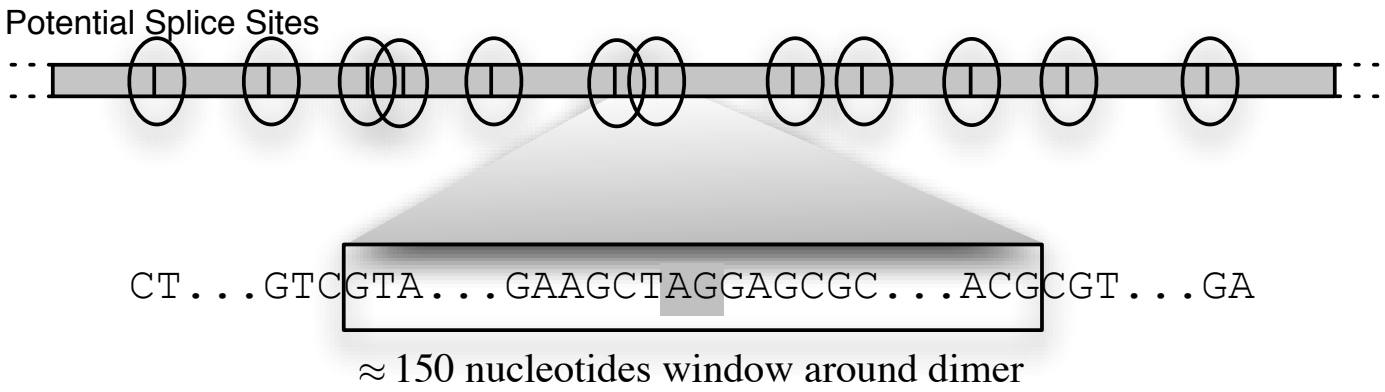
For instance, exploit that exons have higher GC content

or

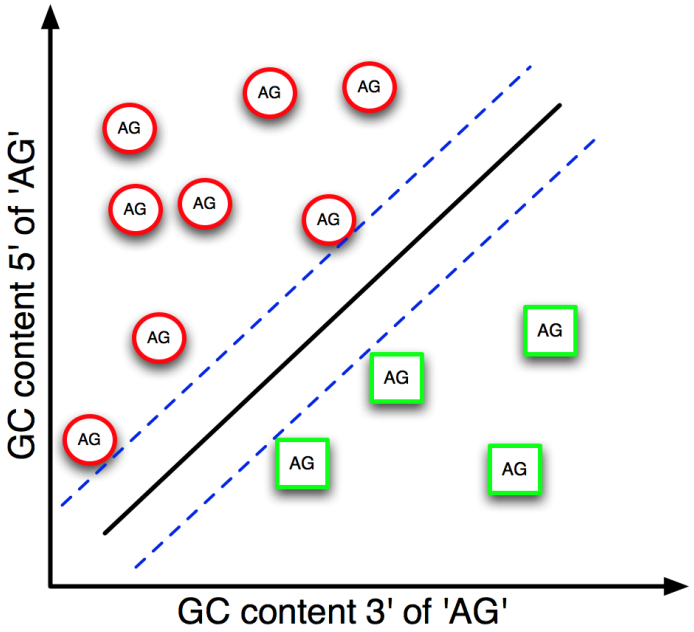
that specific motifs appear near splice sites.

[Sonnenburg et al., 2007b]

Example: Splice Site Recognition



Basic idea:

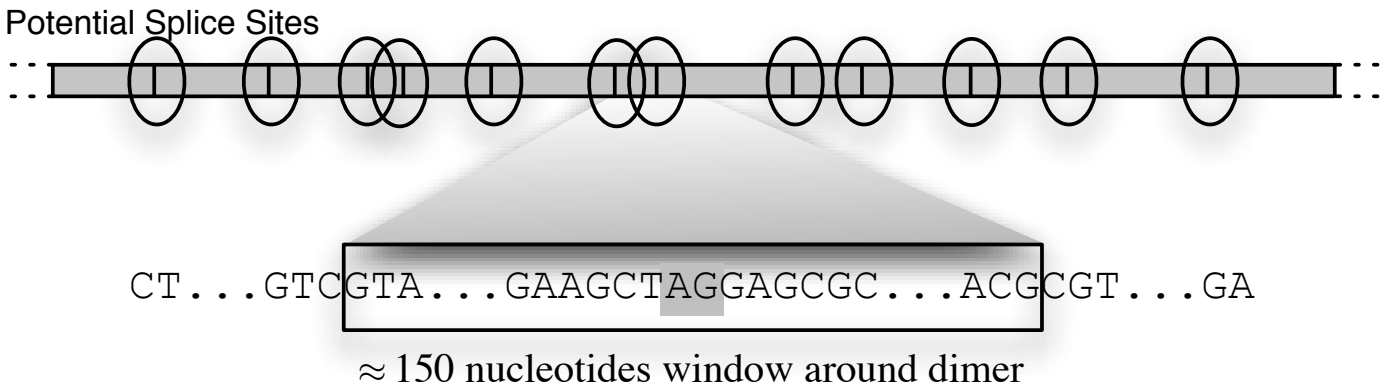


In practice: Use one feature per possible substring (e.g. $\leq$20) at all positions

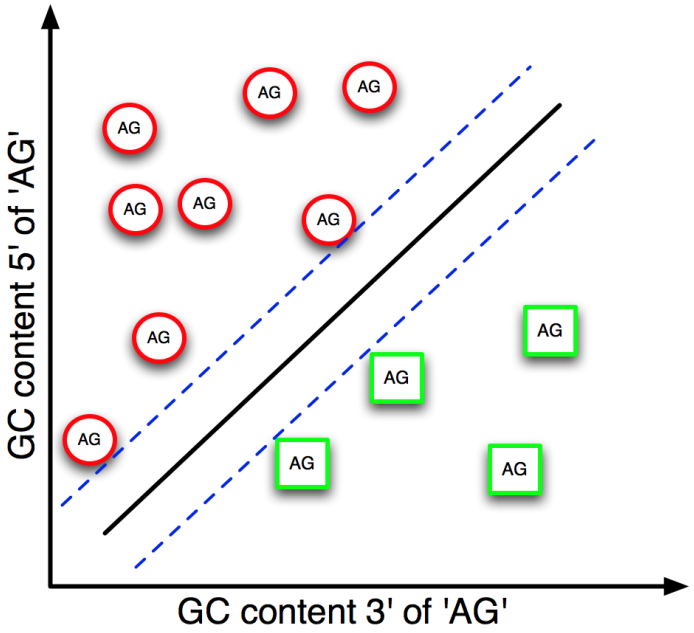
$150 \cdot (4^1 + \ldots + 4^{20}) \approx 2 \cdot 10^{14}$ features

- Needs efficient algorithms
- Leads to most accurate predictors that currently exist

Example: Splice Site Recognition



Basic idea:



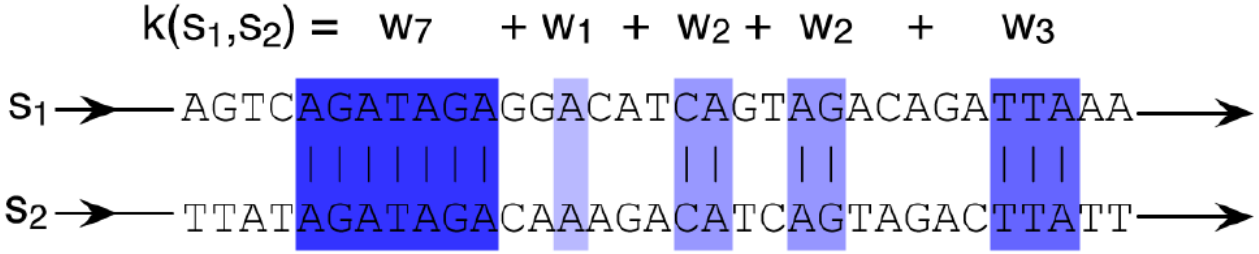
In practice: Use one feature per possible substring (e.g. $\leq$20) at all positions

$150 \cdot (4^1 + \ldots + 4^{20}) \approx 2 \cdot 10^{14}$ features

- Needs efficient algorithms
- Leads to most accurate predictors that currently exist

Kernels and String Data Structures

Luckily, with ℓ_2 regularization, we can use kernels that *implicitly* consider features:



[Rätsch et al., 2007, Rätsch and Sonnenburg, 2004]

Drawback: Training too expensive for millions of examples

Needs more sophisticated algorithms:

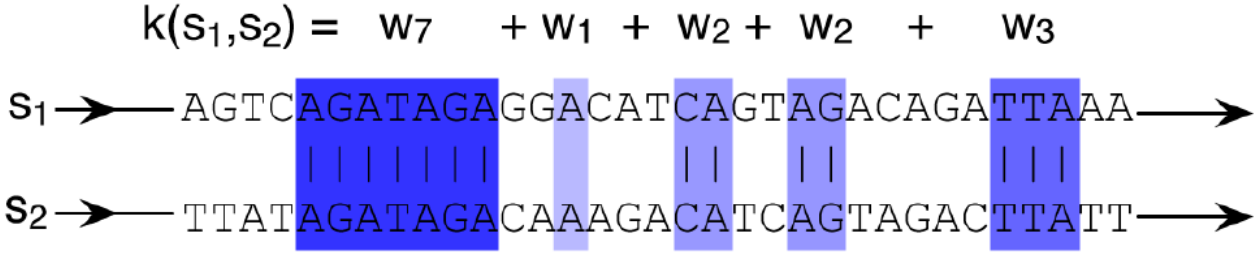
- Exploit that features are sparse and can be explicitly represented
  - String indexing data structures [Sonnenburg et al., 2007a]
- Use novel optimization techniques
  - Use subgradient-based optimization [Franc and Sonnenburg, 2009]

Implemented in a freely available software package

`www.shogun-toolbox.org` [Sonnenburg et al., 2010]

Kernels and String Data Structures

Luckily, with ℓ_2 regularization, we can use kernels that *implicitly* consider features:



[Rätsch et al., 2007, Rätsch and Sonnenburg, 2004]

Drawback: Training is too expensive for millions of examples

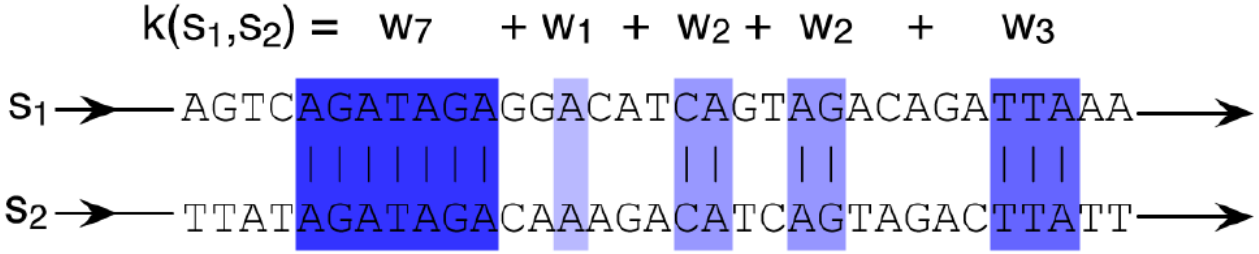
Needs more sophisticated algorithms:

- Exploit that features are sparse and can be explicitly represented
  - String indexing data structures [Sonnenburg et al., 2007a]
- Use novel optimization techniques
  - Use subgradient-based optimization [Franc and Sonnenburg, 2009]

Implemented in a freely available software package

www.shogun-toolbox.org [Sonnenburg et al., 2010]

Luckily, with ℓ_2 regularization, we can use kernels that *implicitly* consider features:



[Rätsch et al., 2007, Rätsch and Sonnenburg, 2004]

Drawback: Training was too expensive for millions of examples

Needs more sophisticated algorithms:

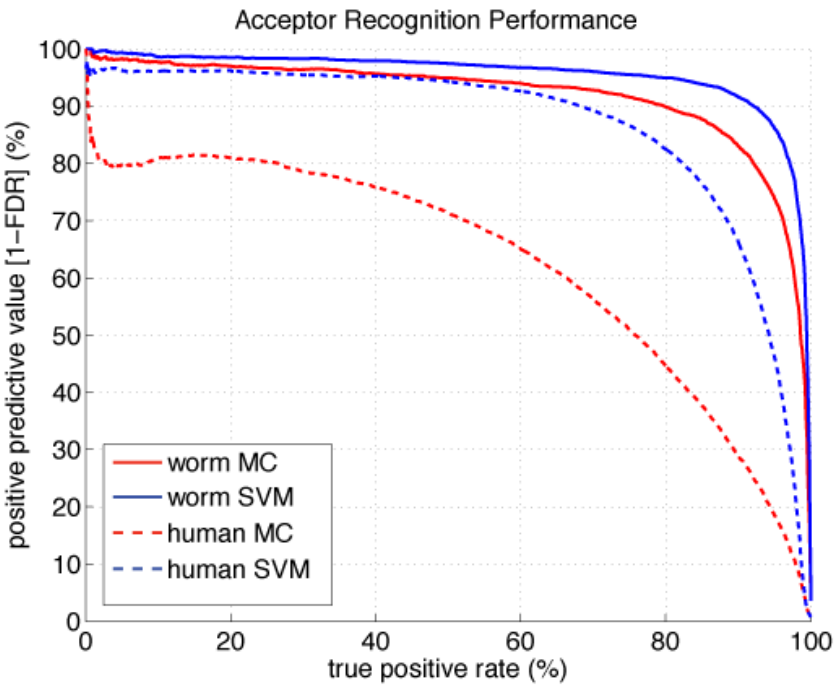
- Exploit that features are sparse and can be explicitly represented
  - String indexing data structures [Sonnenburg et al., 2007a]
- Use novel optimization techniques
  - Use subgradient-based optimization [Franc and Sonnenburg, 2009]

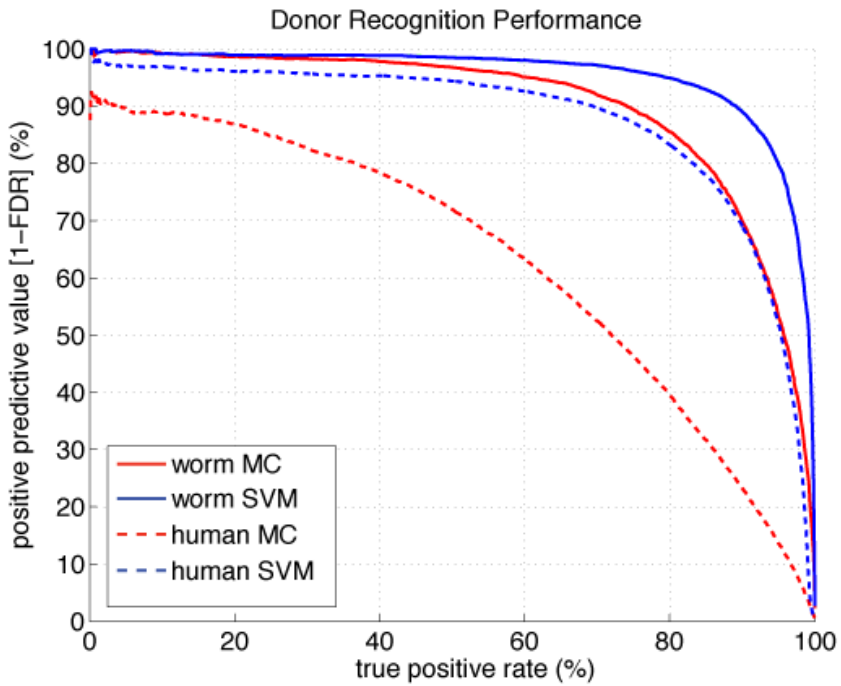
Implemented in a freely available software package

`www.shogun-toolbox.org` [Sonnenburg et al., 2010]

Results on Splice Site Recognition

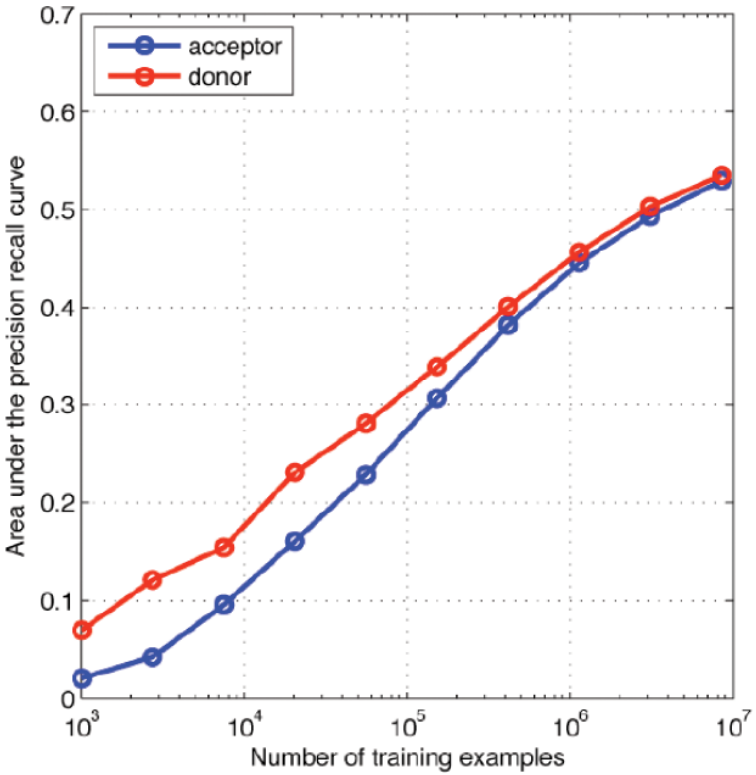
	Worm		Fly		Cress		Fish		Human	
	Acc	Don	Acc	Don	Acc	Don	Acc	Don	Acc	Don
Markov Chain auPRC(%)	92.1	90.0	80.3	78.5	87.4	88.2	63.6	62.9	16.2	26.0
SVM auPRC(%)	95.9	95.3	86.7	87.5	92.2	92.9	86.6	86.9	54.4	56.9

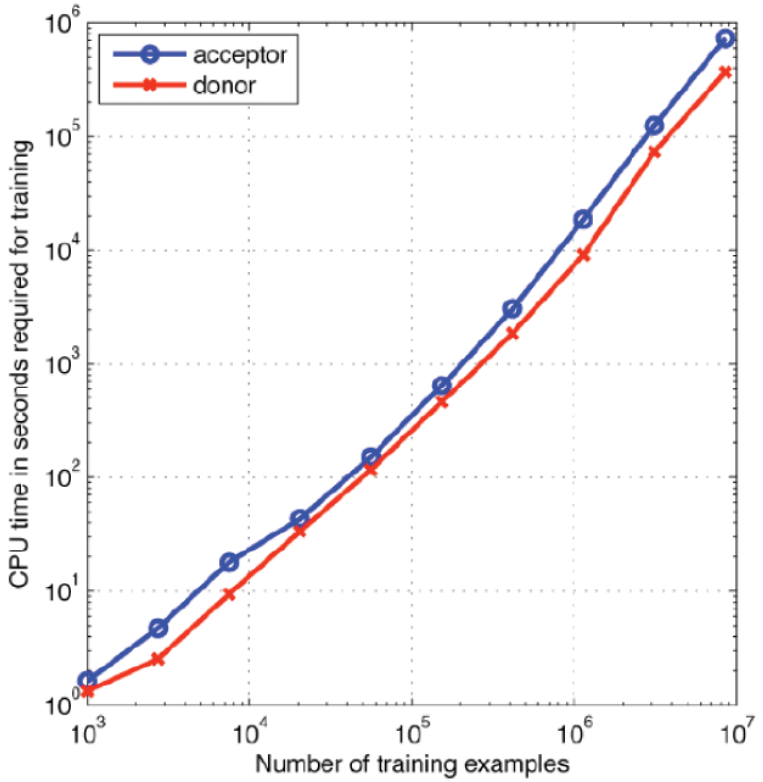




[Sonnenburg, Schweikert, Philips, Behr, Rätsch, 2007]

Human Splice Sites with WD Kernel



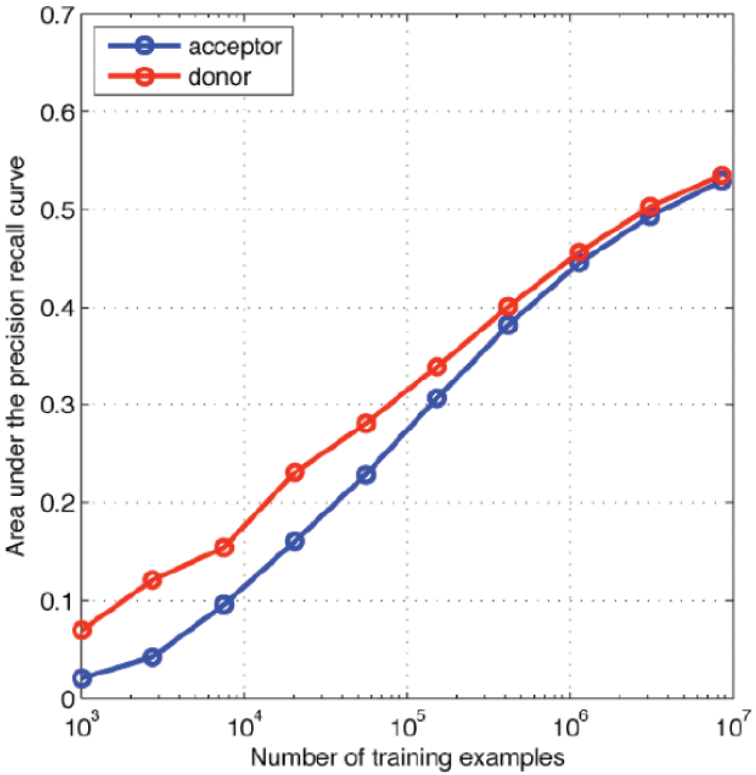


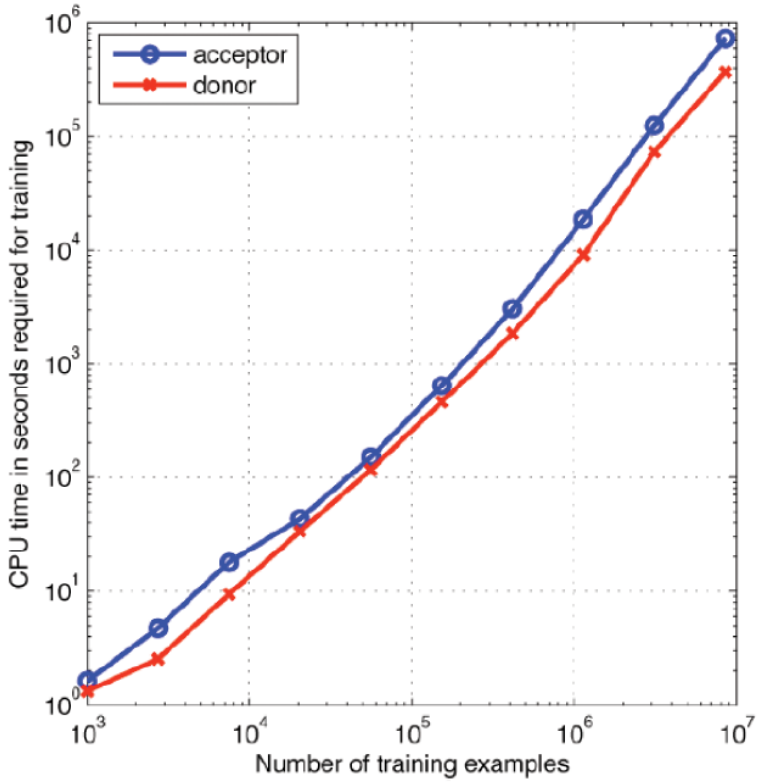
Performance (=Area under PRC curve) keeps increasing.

- Pays off to use all available data!
- Regularizer less important for accuracy when data abundant
- Efficient optimization for quadratic regularizer

[Sonnenburg et al., 2007b]

Human Splice Sites with WD Kernel



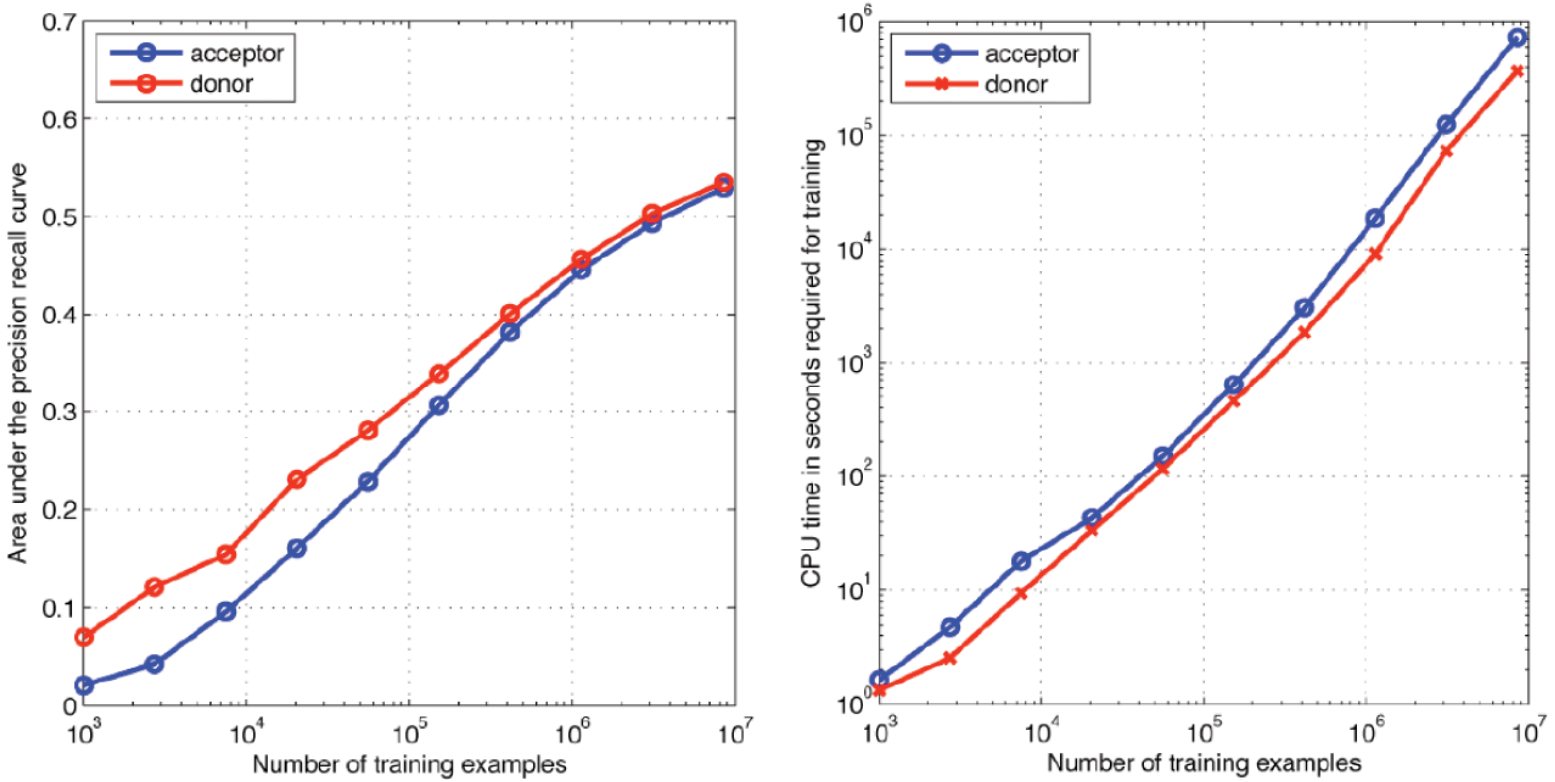


Performance (=Area under PRC curve) keeps increasing.

- Pays off to use all available data!
- Regularizer less important for accuracy when data abundant
- Efficient optimization for quadratic regularizer

[Sonnenburg et al., 2007b]

Human Splice Sites with WD Kernel

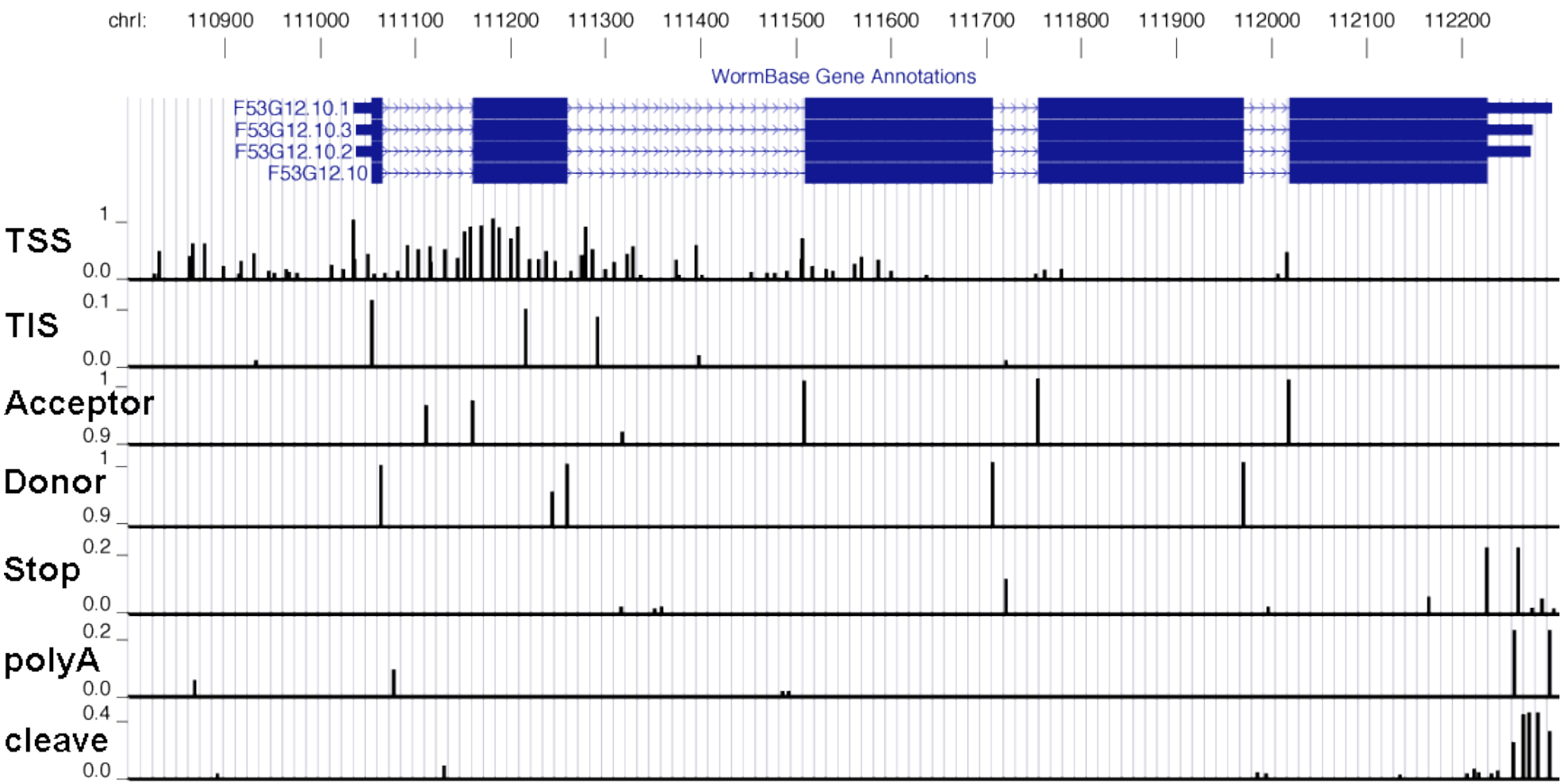


Performance (=Area under PRC curve) keeps increasing.

- Pays off to use all available data!
- Regularizer less important for accuracy when data abundant
- Efficient optimization for quadratic regularizer

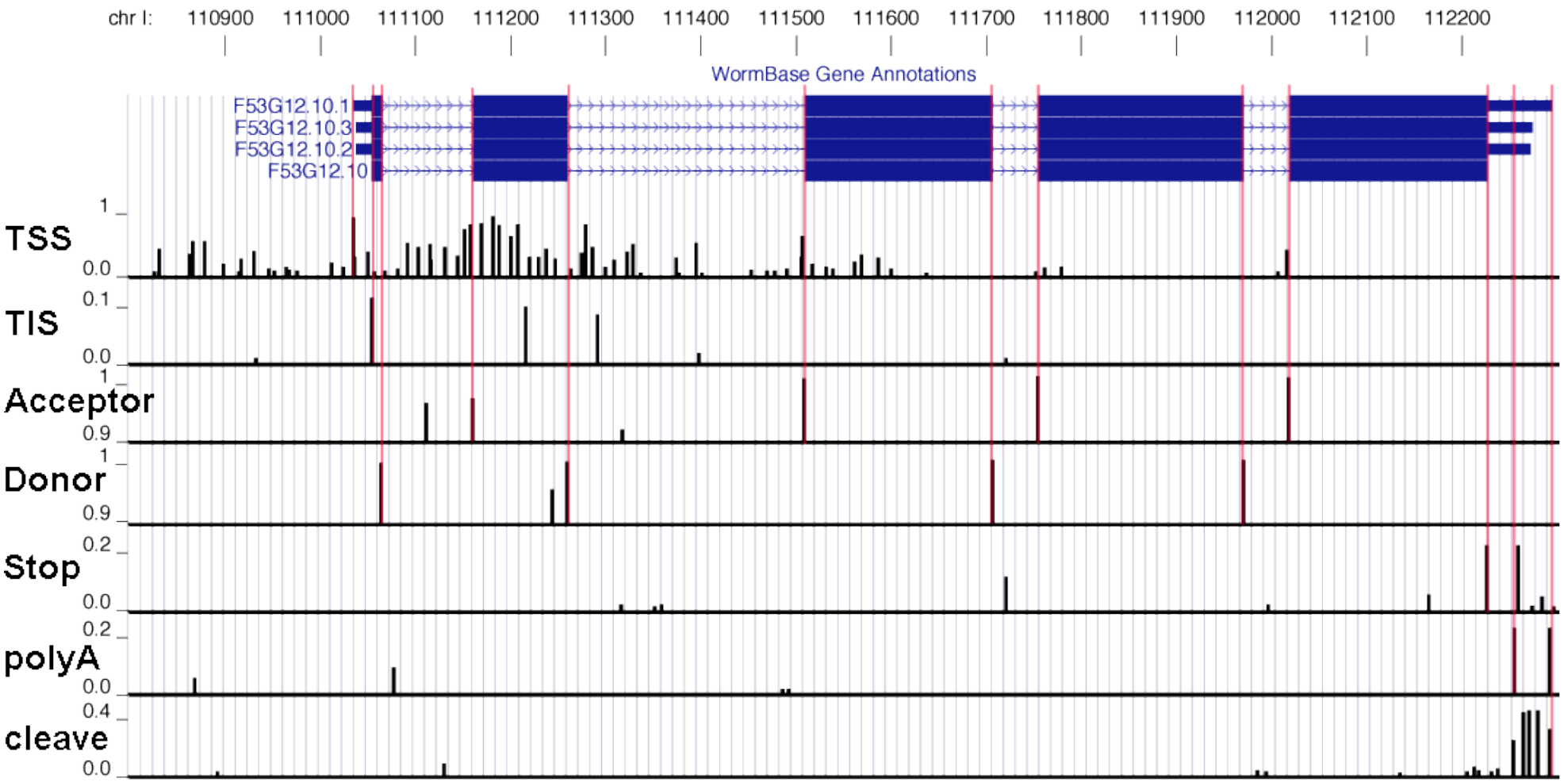
[Sonnenburg et al., 2007b]

Example: Predictions in UCSC Browser



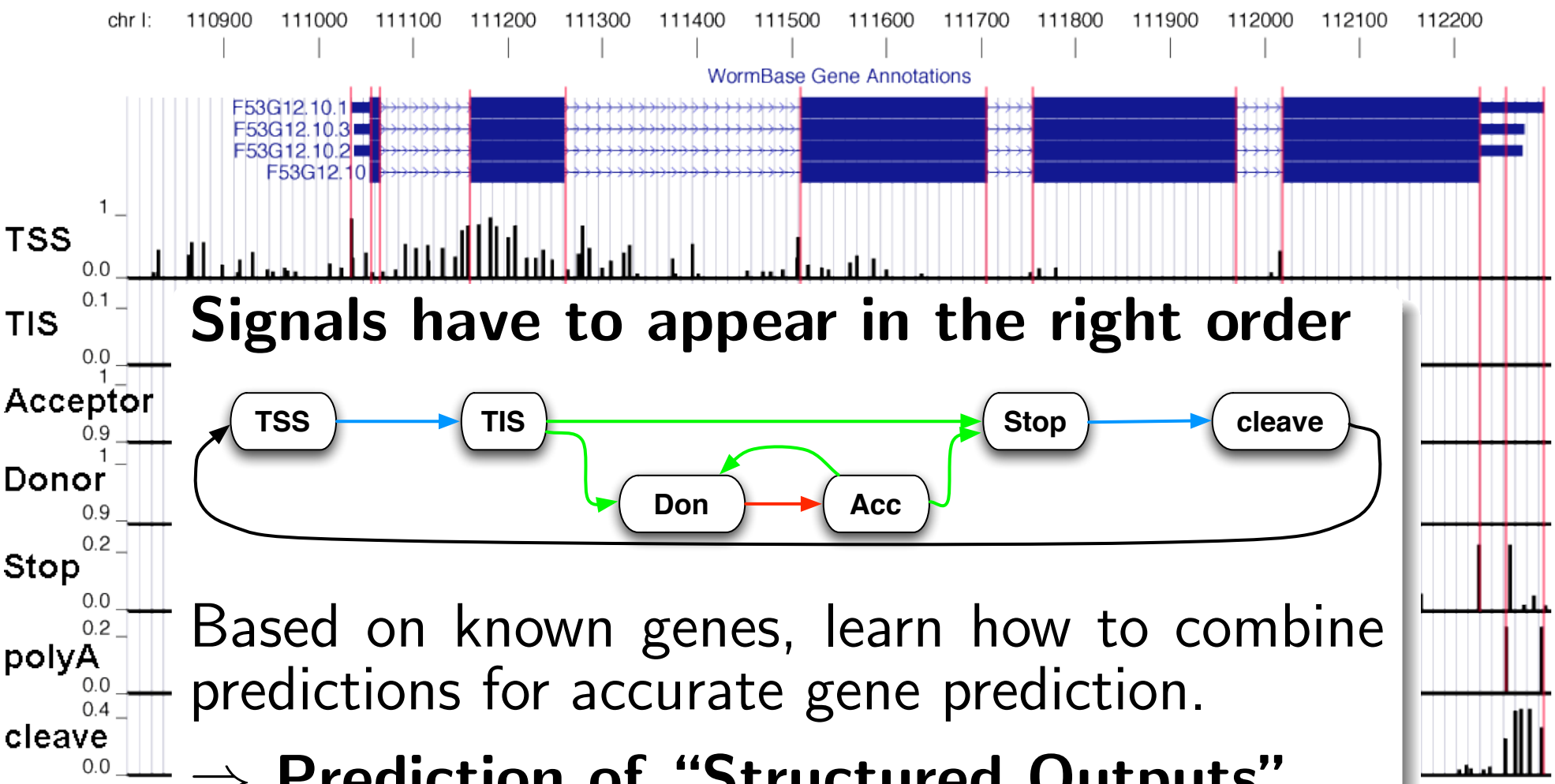
[Rätsch et al., 2007, Schweikert et al., 2009c]

Example: Predictions in UCSC Browser



[Rätsch et al., 2007, Schweikert et al., 2009c]

Example: Predictions in UCSC Browser



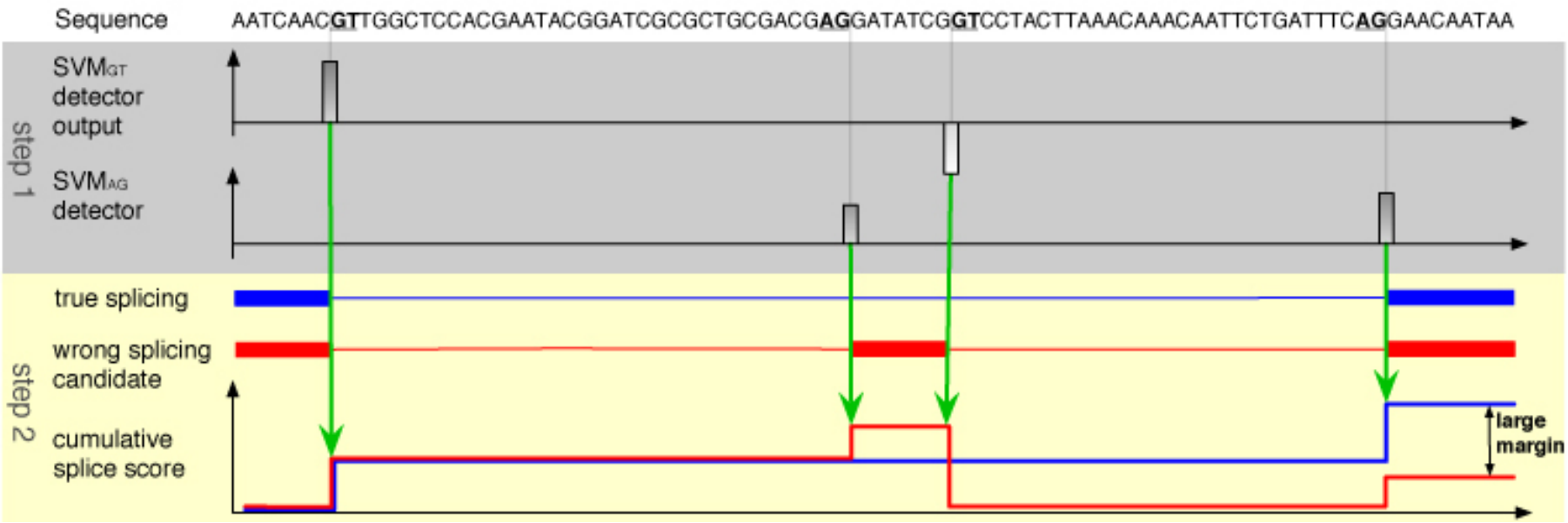
Signals have to appear in the right order

Based on known genes, learn how to combine predictions for accurate gene prediction.

$\Rightarrow$ **Prediction of "Structured Outputs"**

[Rätsch et al., 2007, Schweikert et al., 2009c]

Discriminative Gene Prediction (simplified)



[Rätsch, Sonnenburg, Srinivasan, Witte, Müller, Sommer, Schölkopf, 2007]

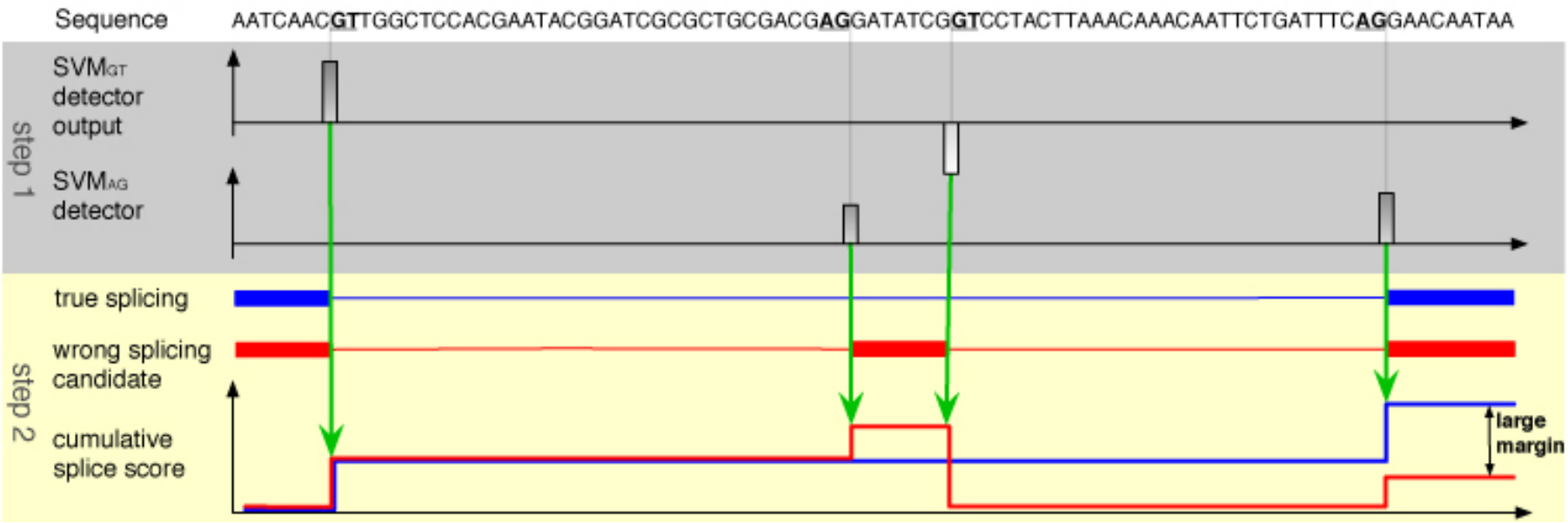
Simplified Model: Score for splice form $\mathbf{y} = \{(p_j, q_j)\}_{j=1}^{J}$:

$$F(\mathbf{y}) := \underbrace{\sum_{j=1}^{J-1} S_{GT}(f_j^{GT}) + \sum_{j=2}^{J} S_{AG}(f_j^{AG})}_{\text{Splice signals}} + \underbrace{\sum_{j=1}^{J-1} S_{L_I}(p_{j+1} - q_j) + \sum_{j=1}^{J} S_{L_E}(q_j - p_j)}_{\text{Segment lengths}}$$

Tune free parameters by solving **structured output problem**.

Discriminative Gene Prediction (simplified)

Friedrich Miescher Laboratory
of the Max Planck Society



[Rätsch, Sonnenburg, Srinivasan, Witte, Müller, Sommer, Schölkopf, 2007]

Simplified Model: Score for splice form $\mathbf{y} = \{(p_j, q_j)\}_{j=1}^{J}$:

$$F(\mathbf{y}) := \underbrace{\sum_{j=1}^{J-1} S_{GT}(f_j^{GT}) + \sum_{j=2}^{J} S_{AG}(f_j^{AG})}_{\text{Splice signals}} + \underbrace{\sum_{j=1}^{J-1} S_{L_I}(p_{j+1} - q_j) + \sum_{j=1}^{J} S_{L_E}(q_j - p_j)}_{\text{Segment lengths}}$$

Tune free parameters by solving **structured output problem**.

Problem-specific Regularizers

- Given the structure of the problem, we can assume that
  - functions transforming signal predictions are monotonic, i.e.

$$f(x) \leq f(x'), \text{if } x \leq x'$$

  - functions should be smooth, e.g. small

$$\int_{x_{\min}}^{x_{\max}} f'(x)^2 \mathrm{d}x \quad \approx \quad \sum_{i=1}^{N-1} \left(\frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \right)^2$$

- We use piece-wise linear functions (PLiFs)
  - Can be linearly parametrized by $f(x) = \langle w, \phi(x) \rangle$
  - Constraints can be easily implemented by $w_i \leq w_{i+1}$
  - Regularization term:

$$\sum_{i=1}^{N-1} (w_{i+1} - w_i)^2$$

Problem-specific Regularizers

- Given the structure of the problem, we can assume that
  - functions transforming signal predictions are monotonic, i.e.

$$f(x) \leq f(x'), \text{if } x \leq x'$$

  - functions should be smooth, e.g. small

$$\int_{x_{\min}}^{x_{\max}} f'(x)^2 \mathrm{d}x \quad \approx \quad \sum_{i=1}^{N-1} \left(\frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \right)^2$$

- We use piece-wise linear functions (PLiFs)
  - Can be linearly parametrized by $f(x) = \langle w, \phi(x) \rangle$
  - Constraints can be easily implemented by $w_i \leq w_{i+1}$
  - Regularization term:

$$\sum_{i=1}^{N-1} (w_{i+1} - w_i)^2$$

Results using mGene

- Most accurate *ab initio* method in the nGASP genome annotation challenge for *C. elegans* [Coghlan et al., 2008]
- Validation of gene predictions for *C. elegans*: [Schweikert et al., 2009c]

	No. of genes	No. of genes analyzed	Frac. of genes w/ expression
New genes	2,197	57	$\approx 42\%$
Missing unconf. genes	205	24	$\approx 8\%$

- Annotation of other nematode genomes: [Schweikert et al., 2009c]

Genome	Genome size [Mbp]	No. of genes	No. exons/gene (mean)	mGene accuracy	best other accuracy
C. remanei	235.94	31503	5.7	**96.6%**	93.8%
C. japonica	266.90	20121	5.3	**93.3%**	88.7%
C. brenneri	453.09	41129	5.4	**93.1%**	87.8%
C. briggsae	108.48	22542	6.0	**87.0%**	82.0%

(Evaluation on exon level)

Results using mGene

- Most accurate *ab initio* method in the nGASP genome annotation challenge for *C. elegans* [Coghlan et al., 2008]
- Validation of gene predictions for *C. elegans*: [Schweikert et al., 2009c]

	No. of genes	No. of genes analyzed	Frac. of genes w/ expression
New genes	2,197	57	$\approx 42\%$
Missing unconf. genes	205	24	$\approx 8\%$

- Annotation of other nematode genomes: [Schweikert et al., 2009c]

Genome	Genome size [Mbp]	No. of genes	No. exons/gene (mean)	mGene accuracy	best other accuracy
C. remanei	235.94	31503	5.7	**96.6%**	93.8%
C. japonica	266.90	20121	5.3	**93.3%**	88.7%
C. brenneri	453.09	41129	5.4	**93.1%**	87.8%
C. briggsae	108.48	22542	6.0	**87.0%**	82.0%

(Evaluation on exon level)

Results using mGene

- Most accurate *ab initio* method in the nGASP genome annotation challenge for *C. elegans* [Coghlan et al., 2008]
- Validation of gene predictions for *C. elegans*: [Schweikert et al., 2009c]

	No. of genes	No. of genes analyzed	Frac. of genes w/ expression
New genes	2,197	57	$\approx 42\%$
Missing unconf. genes	205	24	$\approx 8\%$

- Annotation of other nematode genomes: [Schweikert et al., 2009c]

Genome	Genome size [Mbp]	No. of genes	No. exons/gene (mean)	mGene accuracy	best other accuracy
C. remanei	235.94	31503	5.7	**96.6%**	93.8%
C. japonica	266.90	20121	5.3	**93.3%**	88.7%
C. brenneri	453.09	41129	5.4	**93.1%**	87.8%
C. briggsae	108.48	22542	6.0	**87.0%**	82.0%

(Evaluation on exon level)

Limitations/Extensions

- Gene finding (by far) still not perfect
  - New *ab initio* techniques more accurate
  - Many genes mis-predicted (*C. elegans*: $\approx 50\%$; human: $\approx 80\%$)
  - What is missing?
    - Not enough examples for training?
    - Model complexity?
    - Other information?
- Transcriptome is the result of many complex processes
  - Current methods also ignore other important information:
    - Chromatin structure and methylation patterns
    - RNA structure and processing regulation ...
  - *Ab initio* methods cannot predict alternative transcripts
- New experimental data ...
  - Deep RNA sequencing, DNA/histon modifications, ...

Limitations/Extensions

- Gene finding (by far) still not perfect
  - New *ab initio* techniques more accurate
  - Many genes mis-predicted (*C. elegans*: $\approx 50\%$; human: $\approx 80\%$)
  - What is missing?
    - Not enough examples for training?
    - Model complexity?
    - Other information?
- Transcriptome is the result of many complex processes
  - Current methods also ignore other important information:
    - Chromatin structure and methylation patterns
    - RNA structure and processing regulation ...
  - *Ab initio* methods cannot predict alternative transcripts
- New experimental data ...
  - Deep RNA sequencing, DNA/histon modifications, ...

Limitations/Extensions

- Gene finding (by far) still not perfect
  - New *ab initio* techniques more accurate
  - Many genes mis-predicted (*C. elegans*: $\approx 50\%$; human: $\approx 80\%$)
  - What is missing?
    - Not enough examples for training?
    - Model complexity?
    - Other information?
- Transcriptome is the result of many complex processes
  - Current methods also ignore other important information:
    - Chromatin structure and methylation patterns
    - RNA structure and processing regulation ...
  - *Ab initio* methods cannot predict alternative transcripts
- New experimental data ...
  - Deep RNA sequencing, DNA/histon modifications, ...

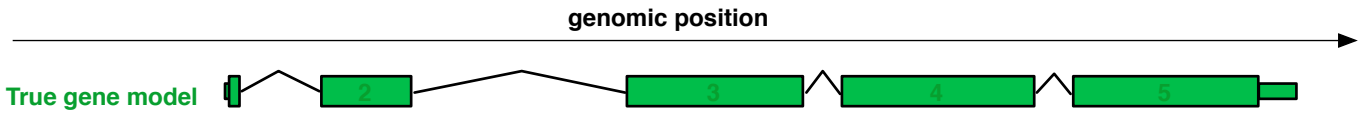
Limitations/Extensions

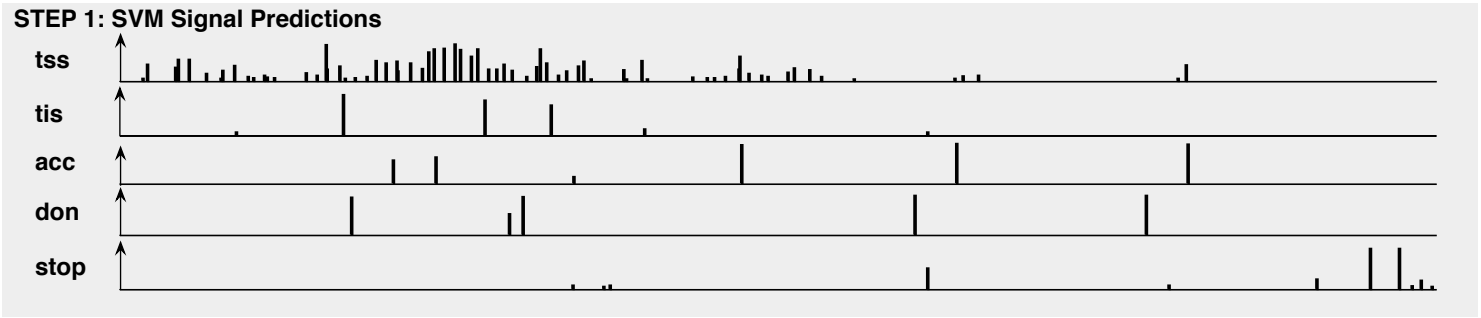
- Gene finding (by far) still not perfect
  - New *ab initio* techniques more accurate
  - Many genes mis-predicted (*C. elegans*: $\approx 50\%$; human: $\approx 80\%$)
  - What is missing?
    - Not enough examples for training?
    - Model complexity?
    - Other information?
- Transcriptome is the result of many complex processes
  - Current methods also ignore other important information:
    - Chromatin structure and methylation patterns
    - RNA structure and processing regulation ...
  - *Ab initio* methods cannot predict alternative transcripts
- New experimental data ...
  - Deep RNA sequencing, DNA/histon modifications, ...

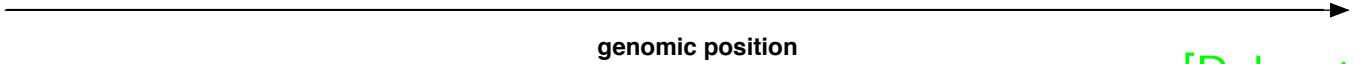
Limitations/Extensions

- Gene finding (by far) still not perfect
  - New *ab initio* techniques more accurate
  - Many genes mis-predicted (*C. elegans*: $\approx 50\%$; human: $\approx 80\%$)
  - What is missing?
    - Not enough examples for training?
    - Model complexity?
    - Other information?
- Transcriptome is the result of many complex processes
  - Current methods also ignore other important information:
    - Chromatin structure and methylation patterns
    - RNA structure and processing regulation ...
  - *Ab initio* methods cannot predict alternative transcripts
- New experimental data ...
  - Deep RNA sequencing, DNA/histon modifications, ...

Learning to Integrate Data Sources

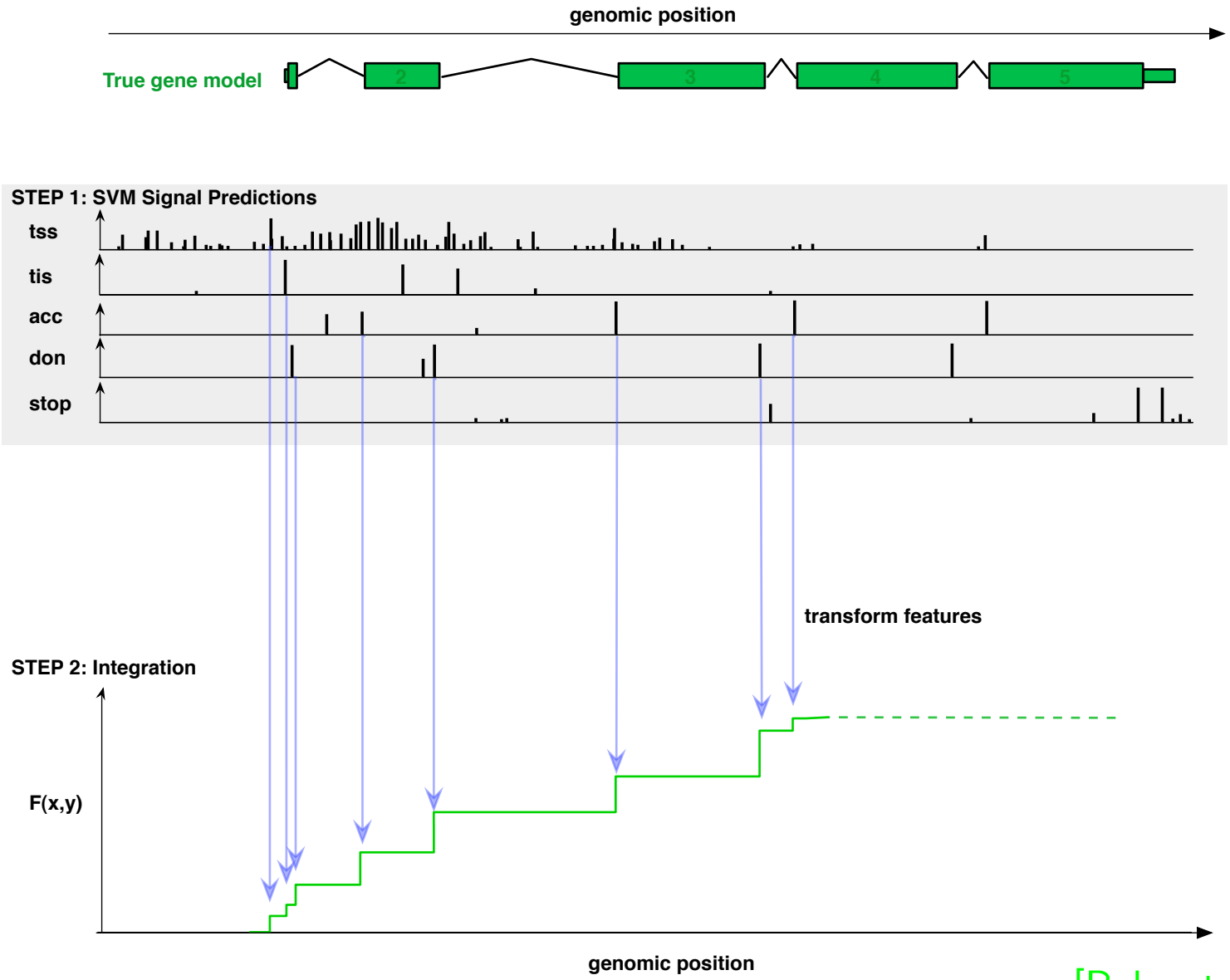






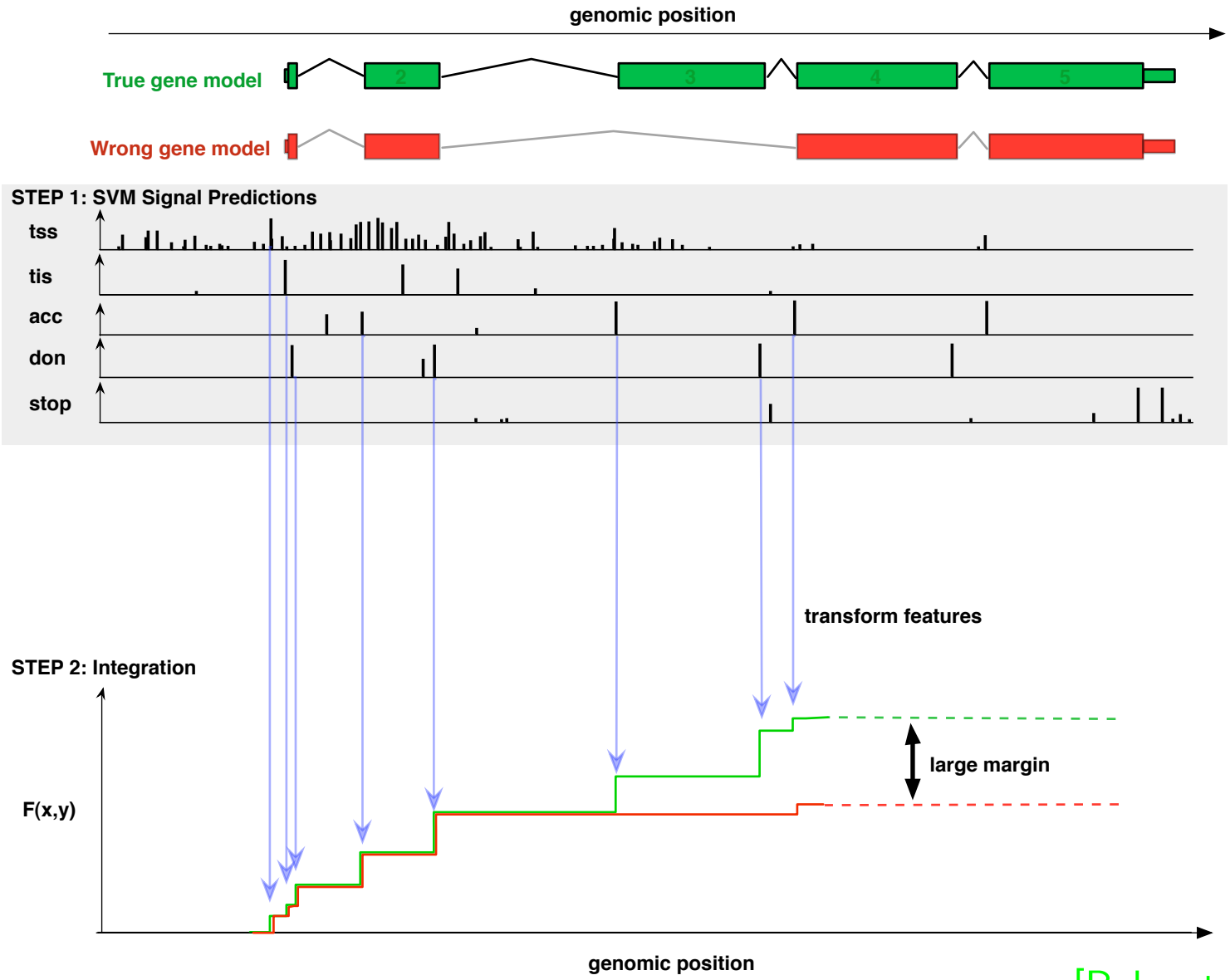
[Behr et al., 2009]

Learning to Integrate Data Sources



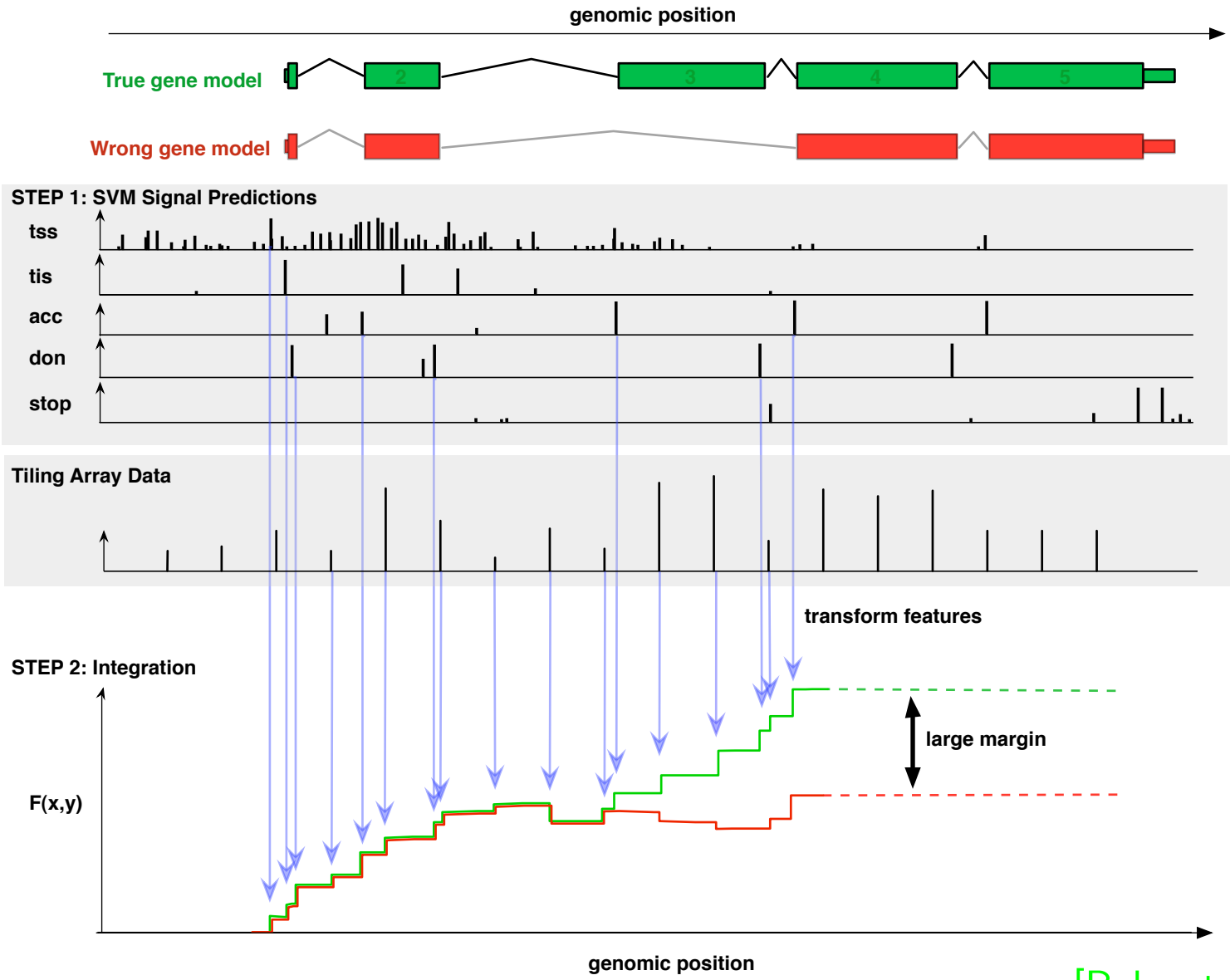
[Behr et al., 2009]

Learning to Integrate Data Sources



[Behr et al., 2009]

Learning to Integrate Data Sources



[Behr et al., 2009]

Regularizer Considerations

Obvious choices:

- In *exonic regions* expression evidence should be scored monotonically increasing
- In *intronic and intergenic regions* expression evidence should be scored monotonically decreasing

More sophisticated for expression-aware model: [Zeller et al., 2008c]

- Replicate whole model $w^{(m)}, m = 1, \ldots, M$ for M expression levels
- Regularize sub-models by coupling them:

$$\sum_{m=1}^{M-1} \sum_{i=1}^{N} (w_i^{(m+1)} - w_i^{(m)})^2$$

Regularizer Considerations

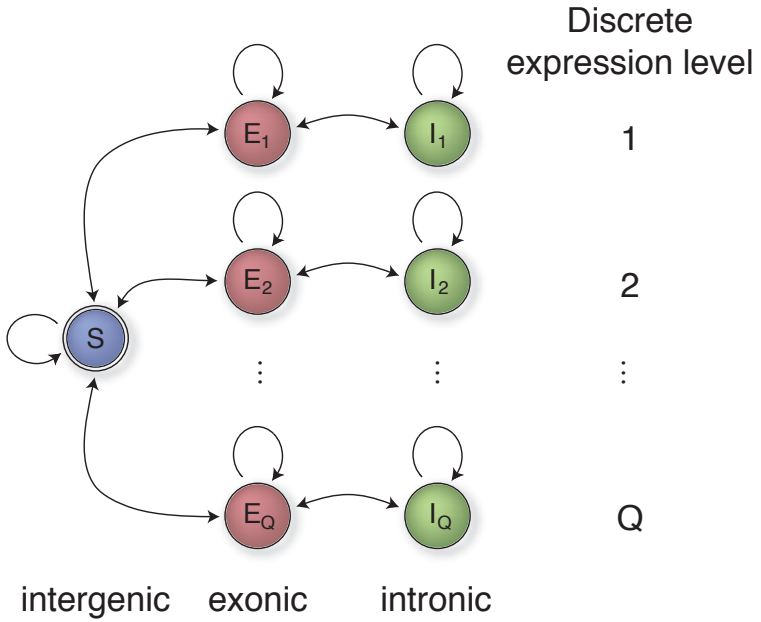
Obvious choices:

- In *exonic regions* expression evidence should be scored monotonically increasing
- In *intronic and intergenic regions* expression evidence should be scored monotonically decreasing

More sophisticated for expression-aware model: [Zeller et al., 2008c]

- Replicate whole model $w^{(m)}, m = 1, \ldots, M$ for M expression levels
- Regularize sub-models by coupling them:

$$\sum_{m=1}^{M-1} \sum_{i=1}^{N} \left(w_i^{(m+1)} - w_i^{(m)} \right)^2$$



Is there other information we can use to improve accuracy?

Study 1: (*A. thaliana*, [Behr et al., 2009]) transcript level $(SN + SP)/2$

1. mGene (*ab initio*) ... **73.3%**
2. ...with DNA methylation (1 tissue) **76.1%**
3. ...with Nucleosome position predictions **78.0%**
4. ...with RNA secondary structure predictions **76.7%**

Study 2: (*C. elegans*, [Behr et al., 2010, in prep.]) transcript level $(SN + SP)/2$

1. mGene (*ab initio*) ... **45.0%**
2. ...with mass spectra ... **45.0%**
3. ...with tiling arrays ... **45.7%**
4. ...with ESTs (influenced annotation) ... **56.5%**
5. ...with RNA-seq ... **55.8%**

Is there other information we can use to improve accuracy?

Study 1: (*A. thaliana*, [Behr et al., 2009]) transcript level $(SN + SP)/2$

1. mGene (*ab initio*) . . . **73.3%**
2. . . . with DNA methylation (1 tissue) **76.1%**
3. . . . with Nucleosome position predictions **78.0%**
4. . . . with RNA secondary structure predictions **76.7%**

Study 2: (*C. elegans*, [Behr et al., 2010, in prep.]) transcript level $(SN + SP)/2$

1. mGene (*ab initio*) . . . **45.0%**
2. . . . with mass spectra . . . **45.0%**
3. . . . with tiling arrays . . . **45.7%**
4. . . . with ESTs (influenced annotation) . . . **56.5%**
5. . . . with RNA-seq . . . **55.8%**

Is there other information we can use to improve accuracy?

Study 1: (*A. thaliana*, [Behr et al., 2009]) transcript level $(SN + SP)/2$

1. mGene (*ab initio*) ... **73.3%**
2. ... with DNA methylation (1 tissue) **76.1%**
3. ... with Nucleosome position predictions **78.0%**
4. ... with RNA secondary structure predictions **76.7%**

Study 2: (*C. elegans*, [Behr et al., 2010, in prep.]) transcript level $(SN + SP)/2$

1. mGene (*ab initio*) ... **45.0%**
2. ... with mass spectra ... **45.0%**
3. ... with tiling arrays ... **45.7%**
4. ... with ESTs (influenced annotation) ... **56.5%**
5. ... with RNA-seq ... **55.8%**

New Genomes

- Adaption to new genomes
  - Requires sufficient number of known gene models for "training"
    - ⇒ Web service [Schweikert et al., 2009b]
- Use *C. elegans* model for other nematodes: [Schweikert et al., 2009c]

Genome	Genome size [Mbp]	No. of genes	No. exons/gene (mean)	mGene accuracy	best other accuracy
C. remanei	235.94	31503	5.7	**96.6%**	93.8%
C. japonica	266.90	20121	5.3	**93.3%**	88.7%
C. brenneri	453.09	41129	5.4	**93.1%**	87.8%
C. briggsae	108.48	22542	6.0	**87.0%**	82.0%

- What if training data is scarce?
  - ⇒ Develop methods that exploit *evolutionary information* and gene models from other genomes [Schweikert et al., 2008]

New Genomes

- Adaption to new genomes
  - Requires sufficient number of known gene models for "training"
    ⇒ Web service [Schweikert et al., 2009b]
- Use *C. elegans* model for other nematodes: [Schweikert et al., 2009c]

Genome	Genome size [Mbp]	No. of genes	No. exons/gene (mean)	mGene accuracy	best other accuracy
C. remanei	235.94	31503	5.7	**96.6%**	93.8%
C. japonica	266.90	20121	5.3	**93.3%**	88.7%
C. brenneri	453.09	41129	5.4	**93.1%**	87.8%
C. briggsae	108.48	22542	6.0	**87.0%**	82.0%

- What if training data is scarce?
  ⇒ Develop methods that exploit *evolutionary information* and gene models from other genomes [Schweikert et al., 2008]

New Genomes

- Adaption to new genomes
  - Requires sufficient number of known gene models for "training"
    - ⇒ Web service [Schweikert et al., 2009b]
- Use *C. elegans* model for other nematodes: [Schweikert et al., 2009c]

Genome	Genome size [Mbp]	No. of genes	No. exons/gene (mean)	mGene accuracy	best other accuracy
C. remanei	235.94	31503	5.7	**96.6%**	93.8%
C. japonica	266.90	20121	5.3	**93.3%**	88.7%
C. brenneri	453.09	41129	5.4	**93.1%**	87.8%
C. briggsae	108.48	22542	6.0	**87.0%**	82.0%

- What if training data is scarce?
  - ⇒ Develop methods that exploit *evolutionary information* and gene models from other genomes [Schweikert et al., 2008]

Hierarchical Transfer Learning

Hierarchical structure arises naturally from the Tree of Life



- Taxonomy used to define relationship between tasks
- Closer tasks benefit more from each other

Formal Problem Definition

Multitask Learning

- Consider M tasks T_i, where $i \in \{1, ..., M\}$
- We are given data $D_i = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{N_i}, y_{N_i})\}$ for each task
- We want to train M predictors $f_1, ..., f_M$, each taking into account all available information
- For that we would like to utilize a given taxonomy $\mathcal{T}$, that relates the tasks at hand

$\Rightarrow$ We need algorithms to efficiently exploit $\mathcal{T}$ for transfer learning

Formal Problem Definition

Multitask Learning

- Consider M tasks T_i, where $i \in \{1, ..., M\}$
- We are given data $D_i = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{N_i}, y_{N_i})\}$ for each task
- We want to train M predictors $f_1, ..., f_M$, each taking into account all available information
- For that we would like to utilize a given taxonomy $\mathcal{T}$, that relates the tasks at hand

$\Rightarrow$ We need algorithms to efficiently exploit $\mathcal{T}$ for transfer learning

Formal Problem Definition

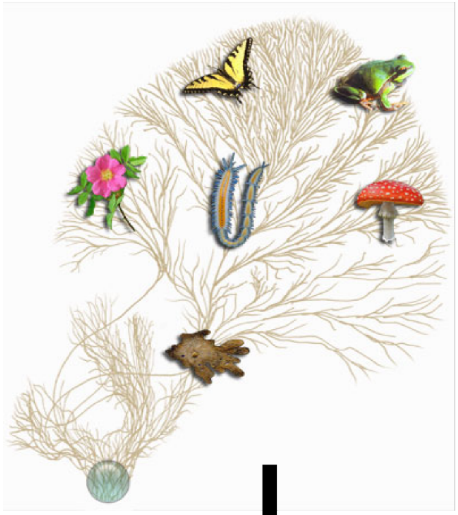
Multitask Learning

- Consider M tasks T_i, where $i \in \{1, ..., M\}$
- We are given data $D_i = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{N_i}, y_{N_i})\}$ for each task
- We want to train M predictors $f_1, ..., f_M$, each taking into account all available information
- For that we would like to utilize a given taxonomy $\mathcal{T}$, that relates the tasks at hand

$\Rightarrow$ We need algorithms to efficiently exploit $\mathcal{T}$ for transfer learning

Two ways of leveraging a given taxonomy $\mathcal{T}$

Taxonomy



Transformation

Task Similarity Matrix

$$\Gamma = \begin{pmatrix} \gamma_{1,1} & \cdots & \gamma_{1,M} \\ \vdots & & \vdots \\ \gamma_{M,1} & \cdots & \gamma_{M,M} \end{pmatrix}$$

Top-Down

Pairwise &

Multitask Kernel

Idea: Enforce model similarity via Regularization Term

- Regular SVM

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{(\mathbf{x},y)\in D} \ell\left(\langle \Phi(\mathbf{x}), \mathbf{w}\rangle + b, y\right)$$

- DA-SVM

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}-\mathbf{w}_{par}\|^2 + C \sum_{(\mathbf{x},y)\in D} \ell\left(\langle \Phi(\mathbf{x}), \mathbf{w}\rangle + b, y\right),$$

where ℓ is the hinge loss, $\ell(z, y) = \max\{1 - yz, 0\}$.

Idea: Enforce model similarity via Regularization Term

- Regular SVM

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{(\mathbf{x},y)\in D} \ell\left(\langle \Phi(\mathbf{x}), \mathbf{w}\rangle + b, y\right)$$

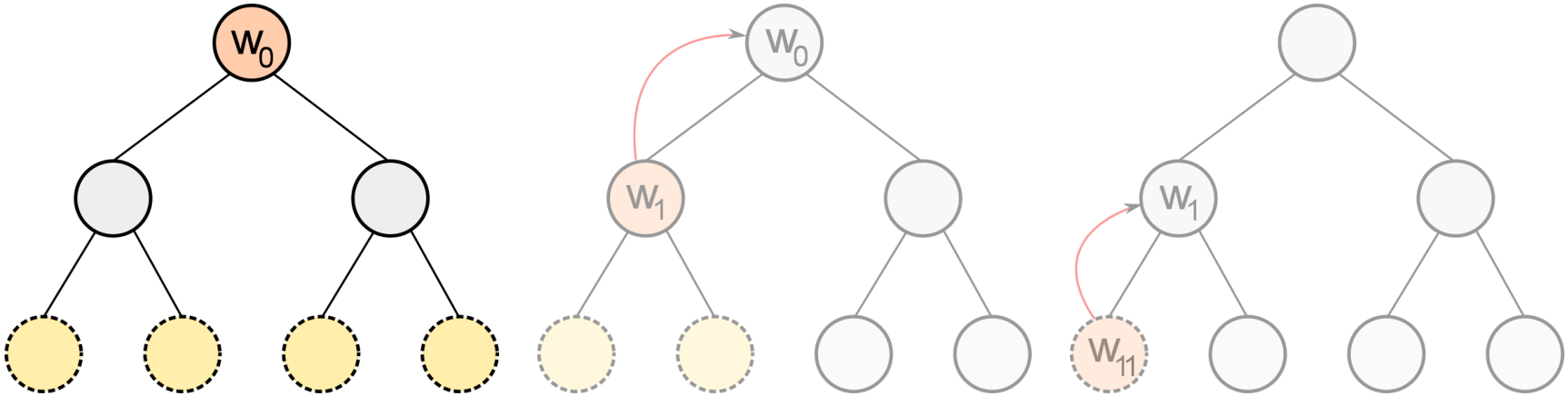
- DA-SVM

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w} - \mathbf{w}_{par}\|^2 + C \sum_{(\mathbf{x},y)\in D} \ell\left(\langle \Phi(\mathbf{x}), \mathbf{w}\rangle + b, y\right),$$

where ℓ is the hinge loss, $\ell(z, y) = \max\{1 - yz, 0\}$.

Idea: Exploit taxonomy G algorithmically

- Initialization: w_0 trained on union of task data
- Top-Down for each node t:
  - Train on $D_i = \bigcup_{j \preccurlyeq i} D_j$
  - Regularize $\mathbf{w}_i$ against parent predictor $\mathbf{w}_{par}$: $\|\mathbf{w}_i - \mathbf{w}_{par}\|^2$
- Use leaf predictors for classification

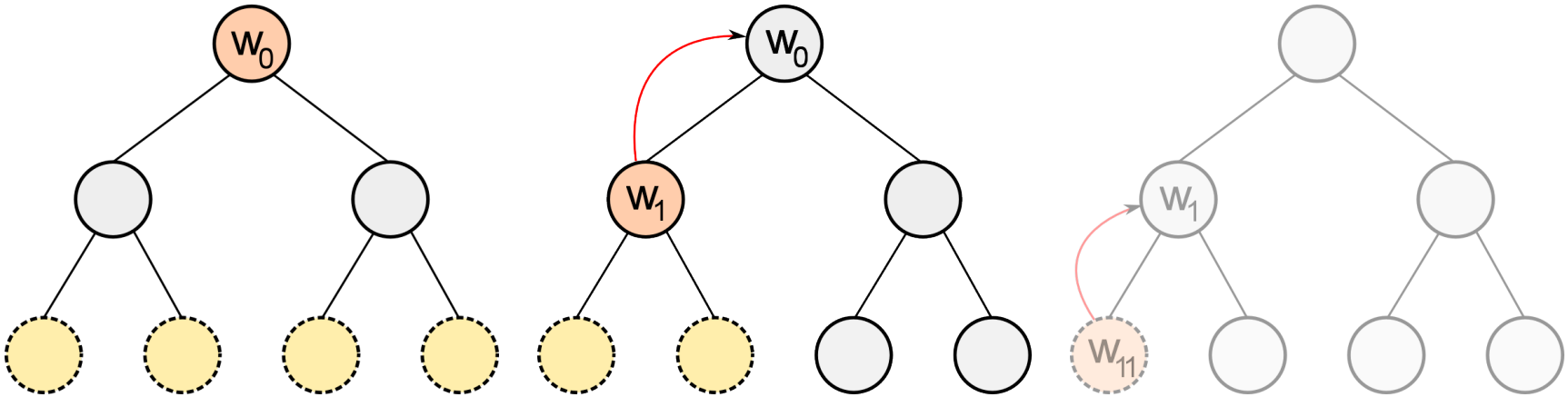


(a) Top-level training (b) Inner training (c) Taxon training

[Widmer et al., 2010]

Idea: Exploit taxonomy G algorithmically

- Initialization: w_0 trained on union of task data
- Top-Down for each node t:
  - Train on $D_i = \bigcup_{j \preccurlyeq i} D_j$
  - Regularize $\mathbf{w}_i$ against parent predictor $\mathbf{w}_{par}$: $\|\mathbf{w}_i - \mathbf{w}_{par}\|^2$
- Use leaf predictors for classification

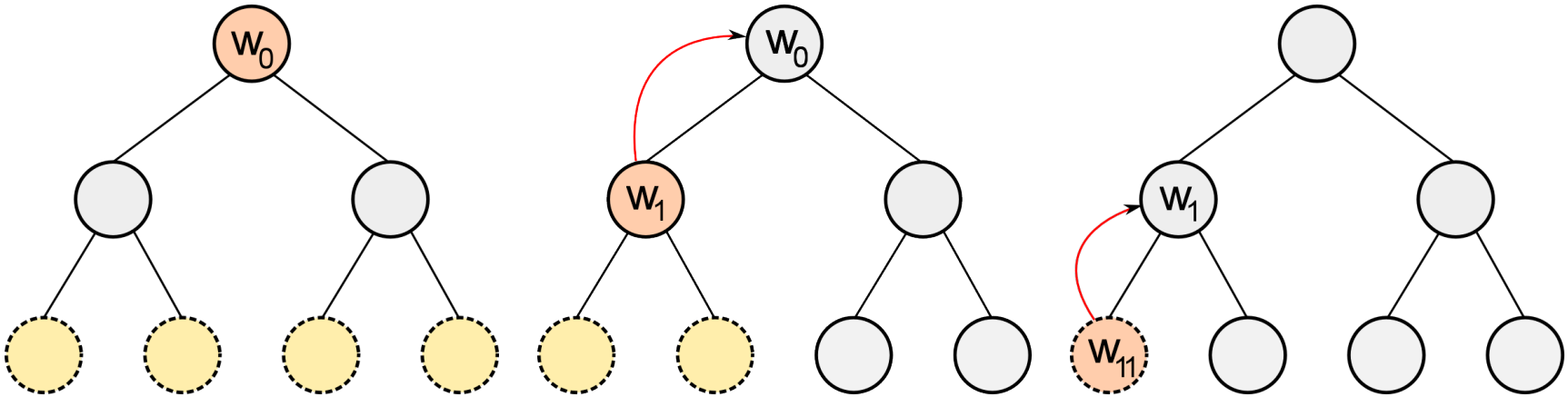


(a) Top-level training (b) Inner training (c) Taxon training

[Widmer et al., 2010]

Idea: Exploit taxonomy G algorithmically

- Initialization: w_0 trained on union of task data
- Top-Down for each node t:
  - Train on $D_i = \bigcup_{j \preccurlyeq i} D_j$
  - Regularize $\mathbf{w}_i$ against parent predictor $\mathbf{w}_{par}$: $\|\mathbf{w}_i - \mathbf{w}_{par}\|^2$
- Use leaf predictors for classification



(a) Top-level training (b) Inner training (c) Taxon training

[Widmer et al., 2010]

Pairwise Approach

$$\min_{\mathbf{w}_1,\ldots,\mathbf{w}_M} \quad \frac{1}{2}\sum_{t=1}^{M}\sum_{s=1}^{M}\gamma_{t,s}\|\mathbf{w}_t-\mathbf{w}_s\|^2+\sum_{t=1}^{M}C_t\sum_{(\mathbf{x},y)\in D_t}\ell\left(\langle\mathbf{x},\mathbf{w}_t\rangle,y\right).$$

where ℓ is the hinge loss, $\ell(z,y)=\max\{1-yz,0\}$.

- Train all classifiers $\mathbf{w}_i$ at the same time
- Loss is evaluated independently on datasets D_i
- Similarity is enforced via pairwise regularization term
- Task closeness controlled by $\gamma_{t,s}$

$\Rightarrow$ efficient solution possible via decomposition

[Evgeniou et al., 2005]

Multitask Kernel Approach

$$\max_{\alpha} -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \hat{k}(\mathbf{x}_i,\mathbf{x}_j) + \sum_{i=1}^{n}\alpha_i$$

$$\text{s.t. } 0 \le \alpha_i \le C \;\forall i \in [1,n]$$

$$\alpha^T \mathbf{y} = 0,$$

where

$$\hat{k}((x_i,s),(x_j,t)) = \underbrace{k_{\text{task}}(s,t)}_{\gamma_{t,s}} \cdot k(x_i,x_j)$$

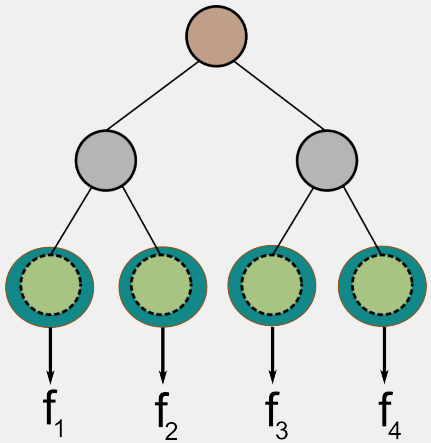
- Easily implemented by altering existing kernel functions (WDK)
- Reuse existing kernel algorithms (SVM)

[Daumé, 2007, Jacob and Vert, 2008, Widmer et al., 2010]

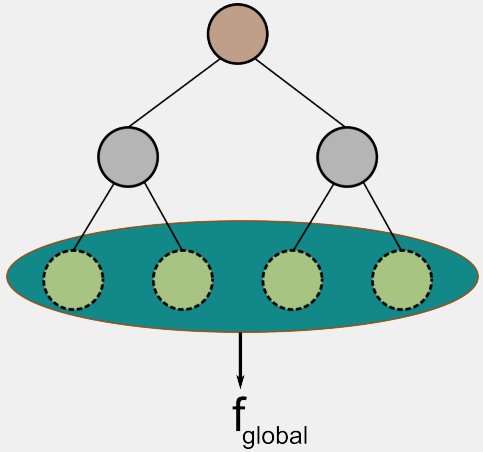
Method Summary

- Multitask Learning Methods
  - Top-Down
  - Pairwise Regularization
  - Multitask Kernel
- Additional Baselines

Plain



Union



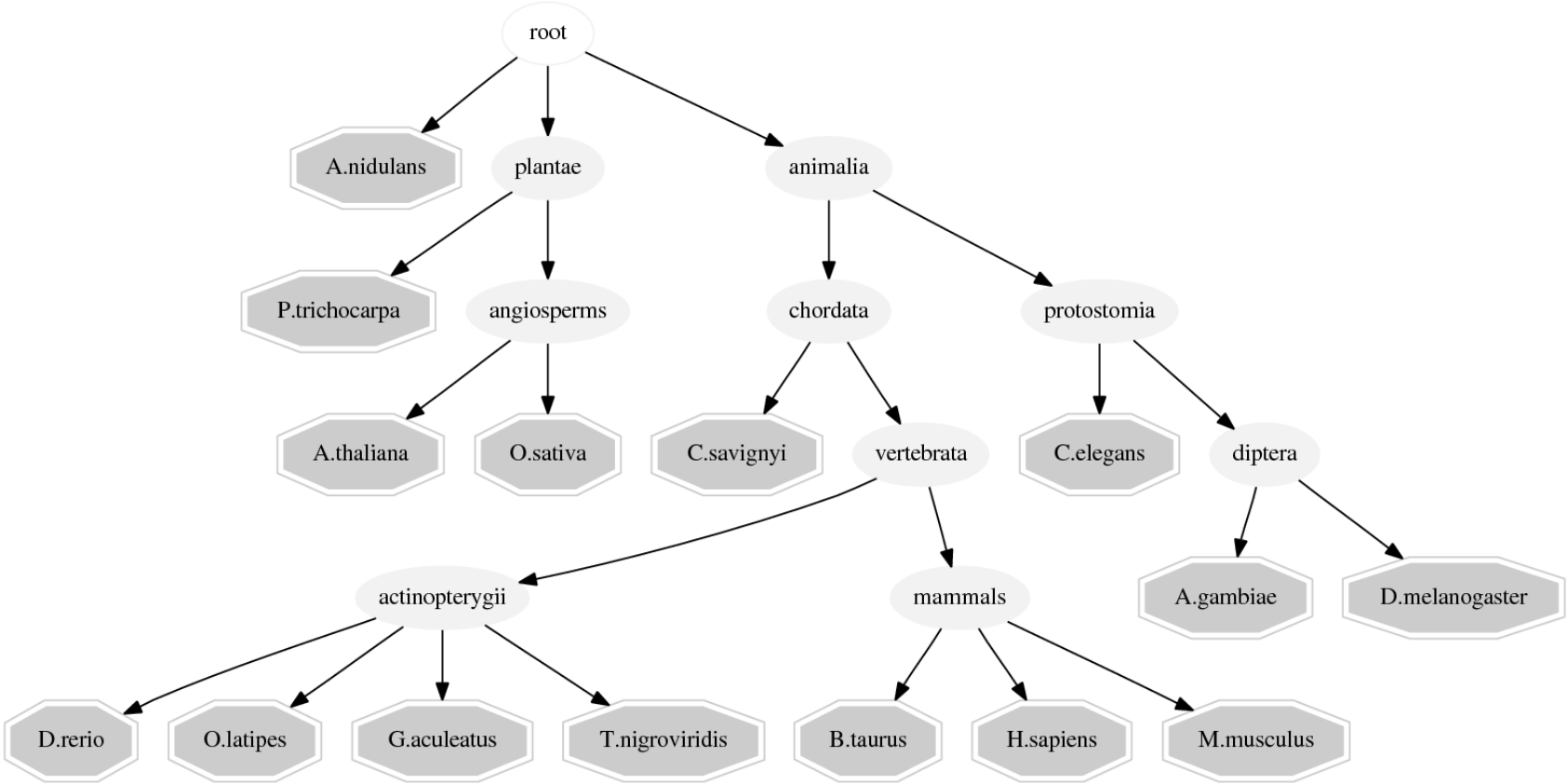
Application to splice-site recognition

- Formulate as binary classification problem

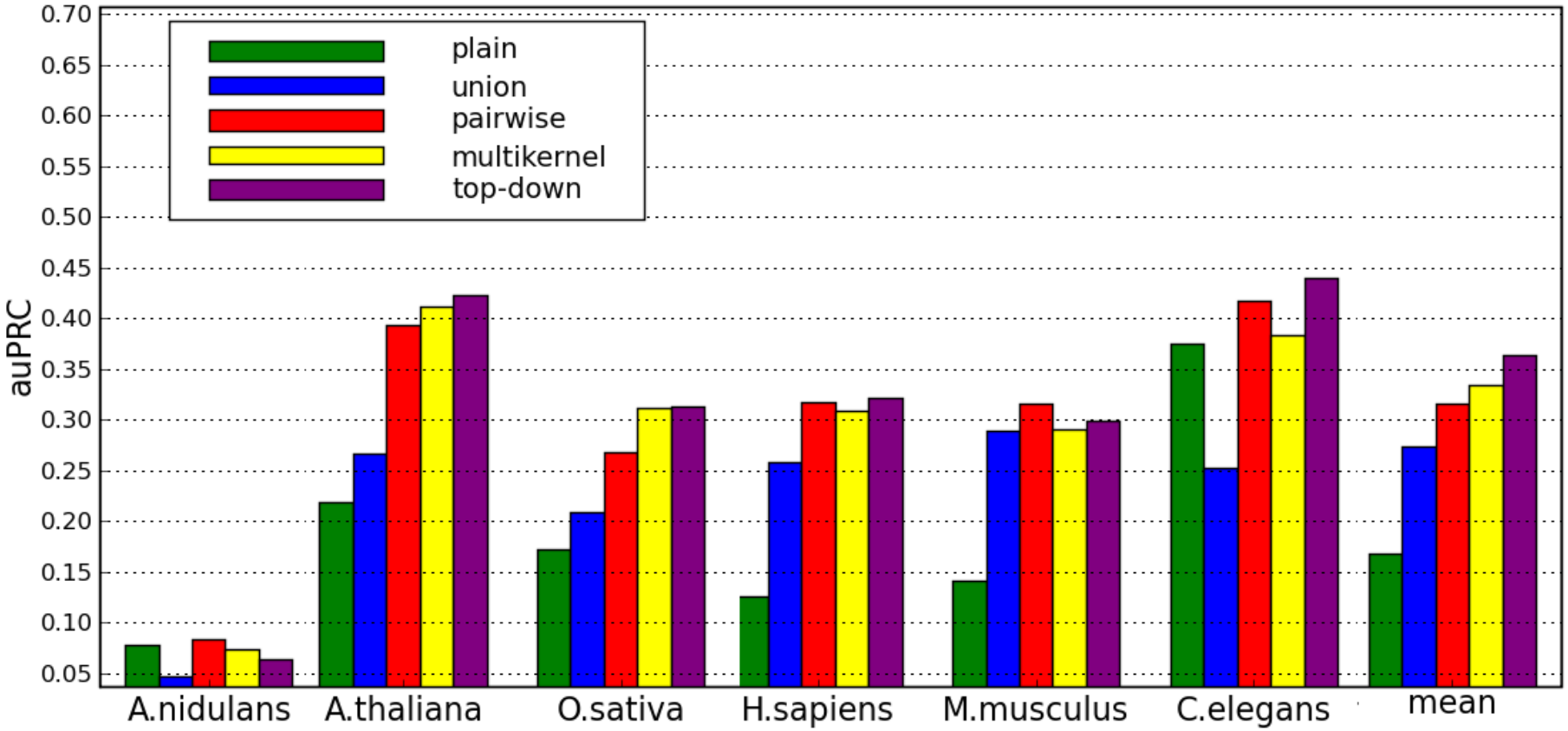
≈ 150 nucleotides window around dimer

CT...GTCGTA...GAAGCTAGGAGCGC...ACGCGT...GA

- Utilize 15 organisms related by taxonomy



Results Splicing Data



- MTL methods outperform baselines
- Best performer is Top-Down
- Potential increase depends on the particular organism

Summary and Future Work

- Regularization
  - Determines convergence rates in optimization
  - Becomes less important, when data is abundant
  - Allows implementation of prior knowledge
  - Helps for solving related learning tasks
- Genome annotation
  - Is a huge *structured output learning problem*, which we solve in a two-step learning procedure
  - Gives rise to heterogeneous data integration challenges
  - Is still quite far from perfect
- Transfer learning
  - Learn models for multiple organisms simultaneously
  - More accurate, but challenges in joint model optimization
- New sequencing data gives
  - Detailed picture on transcriptome
  - Hope to be able to train more sophisticated models

Summary and Future Work

- Regularization
  - Determines convergence rates in optimization
  - Becomes less important, when data is abundant
  - Allows implementation of prior knowledge
  - Helps for solving related learning tasks
- Genome annotation
  - Is a huge *structured output learning problem*, which we solve in a two-step learning procedure
  - Gives rise to heterogeneous data integration challenges
  - Is still quite far from perfect
- Transfer learning
  - Learn models for multiple organisms simultaneously
  - More accurate, but challenges in joint model optimization
- New sequencing data gives
  - Detailed picture on transcriptome
  - Hope to be able to train more sophisticated models

Summary and Future Work

- Regularization
  - Determines convergence rates in optimization
  - Becomes less important, when data is abundant
  - Allows implementation of prior knowledge
  - Helps for solving related learning tasks
- Genome annotation
  - Is a huge *structured output learning problem*, which we solve in a two-step learning procedure
  - Gives rise to heterogeneous data integration challenges
  - Is still quite far from perfect
- Transfer learning
  - Learn models for multiple organisms simultaneously
  - More accurate, but challenges in joint model optimization
- New sequencing data gives
  - Detailed picture on transcriptome
  - Hope to be able to train more sophisticated models

Summary and Future Work

- Regularization
  - Determines convergence rates in optimization
  - Becomes less important, when data is abundant
  - Allows implementation of prior knowledge
  - Helps for solving related learning tasks
- Genome annotation
  - Is a huge *structured output learning problem*, which we solve in a two-step learning procedure
  - Gives rise to heterogeneous data integration challenges
  - Is still quite far from perfect
- Transfer learning
  - Learn models for multiple organisms simultaneously
  - More accurate, but challenges in joint model optimization
- New sequencing data gives
  - Detailed picture on transcriptome
  - Hope to be able to train more sophisticated models

Acknowledgments

Friedrich Miescher Laboratory
of the Max Planck Society

Sequence Analysis

Sören Sonnenburg (FML/FIRST)
Gabi Schweikert (FML/MPI)
Alex Zien (Life Biosystems)
Konrad Rieck (FIRST)

Gene Finding

Jonas Behr (FML)
Gabi Schweikert (FML/MPI)
Georg Zeller (FML/MPI)
Alex Zien (LIFE Biosystems)

Transfer Learning

Christian Widmer (FML)
Jose Leiva (Spain)
Yasemin Altun (MPI)
Bernhard Schölkopf (MPI)

More Information

- http://fml.mpg.de/raetsch
- http://shogun-toolbox.org
- http://mgene.org/web
- Funding from DFG
- Slides with references will be available online

Thank you!

Acknowledgments

Friedrich Miescher Laboratory
of the Max Planck Society

Sequence Analysis

Sören Sonnenburg (FML/FIRST)
Gabi Schweikert (FML/MPI)
Alex Zien (Life Biosystems)
Konrad Rieck (FIRST)

Gene Finding

Jonas Behr (FML)
Gabi Schweikert (FML/MPI)
Georg Zeller (FML/MPI)
Alex Zien (LIFE Biosystems)

Transfer Learning

Christian Widmer (FML)
Jose Leiva (Spain)
Yasemin Altun (MPI)
Bernhard Schölkopf (MPI)

More Information

- `http://fml.mpg.de/raetsch`
- `http://shogun-toolbox.org`
- `http://mgene.org/web`
- Funding from DFG
- Slides with references will be available online

Thank you!

Acknowledgments

Friedrich Miescher Laboratory
of the Max Planck Society

Sequence Analysis

Sören Sonnenburg (FML/FIRST)
Gabi Schweikert (FML/MPI)
Alex Zien (Life Biosystems)
Konrad Rieck (FIRST)

Gene Finding

Jonas Behr (FML)
Gabi Schweikert (FML/MPI)
Georg Zeller (FML/MPI)
Alex Zien (LIFE Biosystems)

Transfer Learning

Christian Widmer (FML)
Jose Leiva (Spain)
Yasemin Altun (MPI)
Bernhard Schölkopf (MPI)

More Information

- `http://fml.mpg.de/raetsch`
- `http://shogun-toolbox.org`
- `http://mgene.org/web`
- Funding from DFG
- Slides with references will be available online

Thank you!

References I

Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden Markov Support Vector Machines. In *Proc. 20th Int. Conf. Mach. Learn.*, pages 3–10, 2003.

J. Behr, G. Schweikert, J. Cao, F. De Bona, G. Zeller, S. Laubinger, S. Ossowski, K. Schneeberger, D. Weigel, and G. Rätsch. Rna-seq and tiling arrays for improved gene finding. Presented at the CSHL Genome Informatics Meeting, May 2008.

J. Behr, G. Schweikert, J. Cao, F. De Bona, G. Zeller, S. Laubinger, S. Ossowski, K. Schneeberger, D. Weigel, and G. Rätsch. Rna-seq and tiling arrays for improved gene finding. Presented at the CSHL Genome Informatics Meeting, July 2009.

RM Clark, G Schweikert, C Toomajian, S Ossowski, G Zeller, P Shinn, N Warthmann, TT Hu, G Fu, DA Hinds, H Chen, KA Frazer, DH Huson, B Schölkopf, M Nordborg, G Rätsch, JR Ecker, and D Weigel. Common sequence polymorphisms shaping genetic diversity in arabidopsis thaliana. *Science*, 317(5836):338–342, 2007. ISSN 1095-9203 (Electronic). doi: 10.1126/science.1138632.

A. Coghlan, T.J. Fiedler, S.J. McKay, P. Flicek, T.W. Harris, D. Blasiar, The nGASP Consortium, and L.D. Stein. ngasp: the nematode genome annotation assessment project. *BMC Bioinformatics*, 2008.

H. Daumé. Frustratingly easy domain adaptation. In *ACL*. The Association for Computer Linguistics, 2007. URL `http://aclweb.org/anthology-new/P/P07/P07-1033.pdf`.

T. Evgeniou, C.A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

References II

V. Franc and S. Sonnenburg. Optimized cutting plane algorithm for large-scale risk minimization. *Journal of Machine Learning Research*, 2009. in press.

L. Jacob and J.P. Vert. Efficient peptide-MHC-I binding prediction for alleles with few known binders. *Bioinformatics*, 24(3):358, 2008.

G. Rätsch and S. Sonnenburg. Accurate splice site detection for *Caenorhabditis elegans*. In K. Tsuda B. Schoelkopf and J.-P. Vert, editors, *Kernel Methods in Computational Biology*. MIT Press, 2004.

G Rätsch and S Sonnenburg. Large scale hidden semi-markov svms. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19, pages 1161–1168, Cambridge, MA, 2007. MIT Press. URL `http://www.fml.tuebingen.mpg.de/raetsch/projects/HSMSVM`.

G. Rätsch and M.K. Warmuth. Efficient margin maximization with boosting. *Journal of Machine Learning Research*, 6:2131–2152, 2005.

G. Rätsch, S. Sonnenburg, and B. Schölkopf. RASE: recognition of alternatively spliced exons in *C. elegans*. *Bioinformatics*, 21(Suppl. 1):i369–i377, June 2005.

G. Rätsch, S. Sonnenburg, J. Srinivasan, H. Witte, K.R. Müller, R. Sommer, and B. Schölkopf. Improving the caenorhabditis elegans genome annotation using machine learning. *PLoS Comput Biol*, 3(2):e20, 2007.

References III

G. Schweikert, G. Zeller, A. Zien, J. Behr, C.S. Ong, P. Philips, A. Bohlen, R. Bohnert, F. De Bona, S. Sonnenburg, and G. Rätsch. mGene: Accurate computational gene finding with application to nematode genomes. under revision, March 2009a.

Gabriele Schweikert, Christian Widmer, Bernhard Schölkopf, and Gunnar Rätsch. An empirical analysis of domain adaptation algorithms. In *Proc. NIPS 2008*, Advances in Neural Information Processing Systems, 2008. accepted.

Gabriele Schweikert, Jonas Behr, Alexander Zien, Georg Zeller, Cheng Soon Ong, Sören Sonnenburg, and Gunnar Rätsch. mgene.web: a web service for accurate computational gene finding. *Nucleic Acids Research*, Web Server Issue, 2009b. URL `http://mgene.org/web`. Advance Access published on June 3, 2009.

Gabriele Schweikert, Alexander Zien, Georg Zeller, Jonas Behr, Christoph Dieterich, Cheng Soon Ong, Petra Philips, Fabio De Bona, Lisa Hartmann, Anja Bohlen, Nina Krüger, Sören Sonnenburg, and Gunnar Rätsch. mgene: Accurate svm-based gene finding with an application to nematode genomes. *Genome Research*, 2009c. URL `http://genome.cshlp.org/content/early/2009/06/29/gr.090597.108.full.pdf+html`. Advance access June 29, 2009.

S. Sonnenburg, G. Rätsch, A. Jagota, and K.-R. Müller. New methods for splice-site recognition. In *Proc. International Conference on Artificial Neural Networks*, 2002.

S. Sonnenburg, G. Rätsch, and K. Rieck. *Large Scale Kernel Machines*, chapter Large Scale Learning with String Kernels. MIT Press, 2007a.

References IV

S Sonnenburg, G Schweikert, P Philips, J Behr, and G Rätsch. Accurate splice site prediction using support vector machines. *BMC Bioinformatics*, 8 Suppl 10:S7, 2007b. ISSN 1471-2105 (Electronic). doi: 10.1186/1471-2105-8-S10-S7.

Sören Sonnenburg, Alexander Zien, and Gunnar Rätsch. ARTS: Accurate recognition of transcription starts in human. *Bioinformatics*, 22(14):e472–480, 2006.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Large-margin methods for structured output spaces. *Journal for Machine Learning Research*, 6:1453–1484, September 2005.

C. Widmer, J. Leiva, Y. Altun, and G. Rätsch. Leveraging sequence classification by taxonomy-based multitask learning. In *Proc. RECOMB'10.*, 2010.

G Zeller, RM Clark, K Schneeberger, A Bohlen, D Weigel, and G Ratsch. Detecting polymorphic regions in arabidopsis thaliana with resequencing microarrays. *Genome Res*, 18 (6):918–929, 2008a. ISSN 1088-9051 (Print). doi: 10.1101/gr.070169.107.

G. Zeller, S.R. Henz, S. Laubinger, D. Weigel, and G Rätsch. Transcript normalization and segmentation of tiling array data. In *Proceedings Pac. Symp. on Biocomputing*, pages 527–538, 2008b.

Georg Zeller, Stefan R Henz, Sascha Laubinger, Detlef Weigel, and Gunnar Rätsch. Transcript normalization and segmentation of tiling array data. *Pacific Symposium on Biocomputing Pacific Symposium on Biocomputing*, pages 527–38, Jan 2008c.

A. Zien, G. Rätsch, S. Mika, B. Schölkopf, T. Lengauer, and K.-R. Müller. Engineering Support Vector Machine Kernels That Recognize Translation Initiation Sites. *BioInformatics*, 16(9): 799–807, September 2000.