A Critical Review of Gene Prediction Software

Mark McElwain

Bioc 218 Final Paper

March 19, 2007

Introduction

In the past several years, there has been a virtual explosion of genomic sequence data with numerous genomes in various stages of sequencing and annotation. In fact, with the number of genomes sequenced numbering over one hundred, it is clear that quick, accurate annotation of predicted genes residing in these genomes is essential to learning more about biology and the evolutionary relationships between species.

In the days of classical, forward genetics, the presence of a gene was inferred from a mutant phenotype, and one could map the mutation to a locus, which suggested the presence of a (usually protein-coding) gene. Now, the wealth of genome data allows investigators to choose a gene based on sequence similarity to an important gene in another organism, for example, or based on the presence of interesting domains. The gene can then be targeted for disruption or, in the case of common model organisms such as the fruit fly *Drosophila melanogaster* or the nematode worm *Caenorhabditis elegans*, a mutation in your gene of interest may already exist in collections of mutations. However, making a guess as to whether or not a specific mutation will have an effect on your gene depends on accurate annotation of the gene and its boundaries. An example of this will appear below.

Reverse genetics these days require accurate annotation of genes and gene boundaries as well. Consider the situation presented below, in which a transposon insertion is identified as causing a specific phenotype, and the researcher wishes to make a deletion in the same region to generated stronger alleles. The decision of whether or not to proceed, and how hopeful he or she should be that a stronger allele will be recovered, also depends upon accurate annotation of gene and exon boundaries.

In this review I will focus on prediction of genes in eukaryotic genomes. Gene prediction in eukaryotes is somewhat more difficult than in prokaryotes, due in part to the decreased gene density in eukaryotic genomes compared to prokaryotic genomes, and in part to the increased complexity of the "gene unit" in eukaryotes compared to prokaryotes.

The structure of a "typical" eukaryotic gene

The very best gene predictor should (in theory) be able to identify the exact boundaries of many different attributes common to most eukaryotic protein-coding genes (Figure 1). Upstream of the protein coding regions, and often many kilobases away lie enhancer regions that often

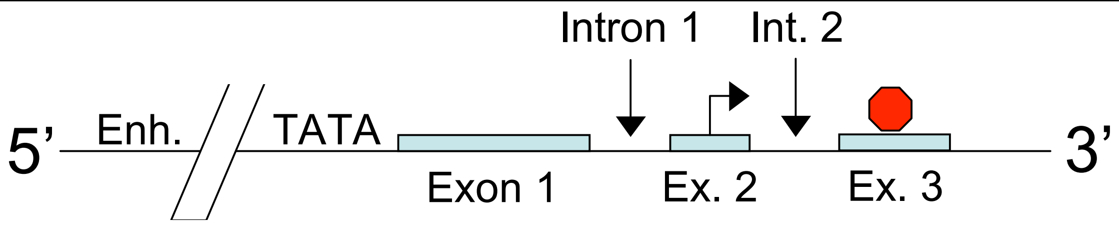

Figure 1. A "typical" eukaryotic gene with upstream enhancer, TATA box, exons, and introns.

confer specificity of gene expression patterns. The distance from the rest of the gene attributes and lack of common sequence motifs likely make this region refractory to prediction by algorithms, and will not be considered in this review. Just upstream of the transcriptional start site is the TATA box, a thymidine and adenosine-rich domain bound by the transcriptional activator complex. Downstream of the transcriptional start siate are the exons, which contribute to the mature mRNA species, and the introns, which are spliced out of the mRNA and are absent in the mature form. Between the introns and exons are specific sequences recognized by the splicing machinery, which can be identified by gene prediction algorithms. Somewhere within an exon also lies the translational start, indicated by an ATG sequence, and somewhere else downstream, also within an exon, lies the translational stop, indicated by a TAG, TAA, or TGA sequence. These sequences should also be able to be identified by gene predictors. Finally, downstream of all exons is the transcriptional stop and polyA tail addition signal.

Many gene prediction programs focus solely on identifying the protein-coding regions of a gene, that is, the region between the start codon and an in-frame stop codon. However, the regions upstream and downstream of the protein coding exons can be very important regulatory regions, and should be included in attempts to define the boundaries of a gene. Indeed, many groups designing these gene detection algorithms re-define an "exon" to mean only a protein-coding exon, which does a disservice to those wishing to understand the boundaries of a gene unit.

Types of gene prediction software

There are two main categories of gene predictors: those that detect sequence motifs and nucleotide content (often called "content sensors" or "*ab initio* gene predictors"), and those that identify regions of high similarity to known coding regions such as cDNAs, expressed sequence tags (ESTs) and annotated protein-coding genes in related organisms. However, it is estimated that less than 50% of genes can be identified solely by similarity to proteins compiled in databases, and cDNAs and ESTs may not cover the entire coding region and may contain errors (Mathe et al., 2002). Using only sequence similarity to sequences in databases may not be the best way to predict the location of genes. However, these methods can be combined with other methods to increase their total predictive power. What follows is a description and analysis of various methods for gene prediction, but is by no means an exhaustive list.

Fgenesh:

Fgenesh is a gene predictor program that falls under the "*ab initio* gene predictor" category (Salamov and Solovyev, 2000). This is a hidden Markov model (HMM)- based program that has parameters for finding genes in human, *Drosophila*, plants, yeast and nematodes. The authors chose to not include information about homology to known cDNAs and ESTs in training their program, as the databases do not have enough coverage to include the entire genome. Therefore, when including cDNA and EST information, the results of a genome-wide gene prediction would be biased toward those genes that have homologs in the databases, and it would be impossible to accurately determine the number of novel genes that can be predicted.

The Fgenesh program was trained on a set of 1600 *Drosophila* genes annotated in GenBank. The program was then tested on 2.9 megabases (Mb) of sequence in the *alcohol dehydrogenase* (*Adh*) region of the *Drosophila* genome, 0.5 Mb at a time. The authors used contiguous sequences rather than overlapping, which could be one source of decreased predicting power. While relatively unlikely, if a gene occurs in this region, features of the gene could be split into two sets of predictions and therefore would not occur together in the input sequence, perhaps making it unlikely that the program could identify it as a gene. However, in this set of sequences there were only 5 of these such breakpoints and are unlikely to greatly affect prediction of the genes encoded (~300-500).

Overall, Fgenesh predicted 1671 exons in 530 genes in this region. The program attained 98% sensitivity [= True Positives (TP) / {TP + False Negatives (FN)}], and 86% specificity [= TP / {TP + False Positives (FP)}] at the nucleotide level, 81% sensitivity and 58% specificity at the exon level, and 72% sensitivity and 39% specificity at the gene level. This suggests that the program does predict some genes that are not annotated as genes, and fails to predict some genes that do exist. It is worth noting that the databases used as the "gold standard" for testing most of these programs are not necessarily 100% true. Therefore, some of the false positives could be true positives missed by the annotators, and some of the false negatives could actually be non-genes and were annotated mistakenly.

The above analysis reveals the quality of the *ab initio* gene prediction of the Fgenesh program. However, the authors do note that there was a set of genes not predicted that can be predicted by including information about homologous protein sequences in Fgenesh. This new program is called Fgenesh+. Within this set of missed genes, Fgenesh+ performed significantly better than Fgenesh in terms of sensitivity and specificity, suggesting that, while ignoring similarity to cDNAs, ESTs, and protein sequences may be appropriate for analyzing the *ab initio* part of a predictor algorithm, for true-life scenarios of predicting genes in newly sequenced eukaryotic genomes, more genes can be predicted by inclusion of these database sequences.

Glimmer:

The Glimmer program was first designed to predict genes in prokaryotic genomes (Salzberg et al., 1998). The program utilized an interpolated Markov model (IMM) to identify sequences likely to encode proteins. The authors found that using this method, which takes into account predictions from 1st – 8th-order Markov models rather than a fixed-order Markov model, outperformed a fixed- (5th-) order Markov model in gene prediction.

Glimmer was then used as a basis for the design of GlimmerM, a method for predicting genes in the genome of the eukaryotic malaria parasite *Plasmodium falciparum* (Salzberg et al., 1999). The genome of *P. falciparum* is estimated to be approximately 20% coding (compared to ~90% for prokaryotes). Humans and other higher eukaryotes are predicted to have an even lower gene density within their genomes, so a program to predict genes in a lower density background was a step in the right direction. GlimmerM included an algorithm for predicting splice sites, where the original Glimmer had none (as prokaryotic genes do not have introns). The program assigned a score to 16 bases surrounding a predicted splice donor sequence, and a score to the 29 bases surrounding a predicted splice acceptor sequence.

Further improvements to GlimmerM for the purposes of eukaryotic gene prediction in eukaryotes were reported in 2004 (Majoros et al., 2004). Like GlimmerM, GlimmerHMM (as it is called) utilizes an IMM. GlimmerHMM also adds in splice site predictors adapted from the GeneSplicer program.

GlimmerHMM was tested on *Arabidopsis thaliana* genes confirmed by the presence of cDNAs (Majoros et al., 2004). The authors found that GlimmerHMM performed with 71% specificity and 79% sensitivity. It should be noted that gene density in the *Arabidopsis* genome is estimated to be around 20%, similar to the genome of *P. falciparum*, so significant improvements are likely to be needed for this program in order to be useful for gene prediction in the genomes of higher eukaryotes. It would be very interesting to directly compare this program to others using the same genomic sequence.

GeneMark:

GeneMark is a program utilizing a Markov model to identify genes in bacterial genomes. First reported in 1993, the main improvement in GeneMark over previous gene prediction programs was simultaneous prediction of genes on both sense and antisense strands of the DNA, rather than scanning one strand after the other (Borodovsky and McIninch, 1993). This method allows decreased false positives from predicting genes on one strand when the real gene is on the complementary strand.

The GeneMark method was later improved, partially by including information about ribosome binding site predictions (Lukashin and Borodovsky, 1998). In theory, this should increase the predictive power of the program, as the more information about features common to every protein-coding gene included in a prediction, the more likely a prediction is to be correct. Indeed, the authors found that the new GeneMark.hmm performed better than the previous version. When tested on the *E. coli* genome, GeneMark.hmm exactly predicted 71% of genes while GeneMark only exactly predicted 62%. GeneMark.hmm also only missed 10% of the genes, while GeneMark missed 13%.

GeneMark was further improved by addition of parameters allowing detection of genes in eukaryotic and viral genomes (Besemer and Borodovsky, 2005). Included in the program (in the eukaryotic version) are predictors for splice sites, translation initiation signals and exons and introns. This method, because of the increased number of features recognized, should represent a significant improvement over previous gene finders. The authors did not assess the specificity or sensitivity of this method, and a comparison between this and others on a eukaryotic genome is essential.

Grail:

The goal of the GRAIL program is to utilize several algorithms detecting different features of a protein coding gene to predict with high accuracy the position of a gene within a genome (Uberbacher et al., 1996). Originally, GRAIL examined the presence of these several features (discussed below) in a sliding 99-nucleotide window; however, this biases the program towards prediction of longer exons and misses a larger number of shorter exons. This bias was later removed by allowing the program to examine all possible exons, rather than just those in a sliding window. In both cases, GRAIL utilizes a neural network to combine predictions for all these gene features.

GRAIL starts by scoring a region as protein coding versus protein noncoding based on frequency of 6-mers that occur often in coding versus noncoding sequences (Uberbacher et al., 1996). These coding regions are then scored for the presence of a start codon, with a stop codon downstream and in-frame. A higher score is achieved by the presence of these features. The GRAIL algorithm can also identify frameshift mutations (insertions or deletions) that may be introduced do to errors during sequencing, by determining when a shift in frame occurs in a region with high coding potential, creating an out-of-frame stop codon. Splice sites are also detected, by analysis of the coding region surrounding splice donor sequences and splice acceptor sequences. GRAIL also scores CpG islands, which are underrepresented in the genome but enriched just 5' of coding regions, the presence of a TATA box, and the polyadenylation signal.

At first glance, GRAIL should be a very fine gene predictor, as it utilizes algorithms for detecting many features common to most protein-coding genes. When the authors tested the program on a region of the human genome, they found that it could correctly identify 21 out of the 30 exons actually present in the sequence. As with the other programs, it is important to compare the accuracy to predictions of other predictors on the same genomic sequence in order to determine its predicting power.

Summary and comparison of gene predictors

Several groups have compared the accuracy of gene predictions by various programs. While no group has the time or resources to test every gene predictor program in existence, testing a subset of programs, all on the same test sequence, gives us a good idea of the differences in their predictive power. Burset and Guigo compared several programs including the abovementioned FGENEH and GRAIL 2 on a set of 570 sequences (Burset and Guigo, 1996). They found that, at the nucleotide level, FGENEH performed with sensitivity = 0.77 and specificity = 0.88, while GRAIL 2 performed with sensitivity = 0.72 and specificity = 0.87. At the exon level, FGENEH performed with sensitivity = 0.61 and specificity = 0.64, while GRAIL 2 performed with sensitivity = 0.36 and specificity = 0.43. At both the nucleotide and exon levels, GRAIL 2 found more false positives and false negatives than FGENEH. This is perhaps surprising, as the GRAIL program takes into account more gene features than FGENEH and should be able to decrease the number of false positives by scoring sequences lacking common gene features as non-coding. However, this may imply that a simpler program only predicting minimal gene elements may perform better because it is less complicated, and there may be a small set of gene features sufficient to predict the presence of a gene, and taking into account more features may just interfere with predictions.

It is a difficult task to think about the differences between gene prediction algorithms. One problem in directly comparing them is that each group often has a different goal for their program, and they are often trained on sequences from different organisms. For example, one group may desire to predict with a high probability that a particular nucleotide or hexamer of nucleotides in the *Drosophila* genome is coding or non-coding, while another group may want to focus on the presence of splice sites in a sequence from the human genome. These programs will be likely trained on different "gold standard" sequences and probably cannot be directly compared. It would be extremely interesting if, in the future, bioinformaticists put effort into training programs to be used on a wide variety of organisms.

Combinations of gene prediction software

Most gene prediction algorithms have unique sets of strengths and weaknesses, and it would follow that combining the predictions from these algorithms in various ways could improve the strength of a prediction. For example, if two or more methods both predict the presence of a gene, it is more likely that this gene is real; likewise, if several methods all fail to predict a gene in a region, it is extremely unlikely that a gene resides in this region in real life.

Pavlovic and colleagues combined the Fgenes, Genie EST, and HMM Gene programs in different ways and tested the resulting combination programs on a 2.9 Mb sequence from the

Drosophila genome (Pavlovic et al., 2002). They found that, at the nucleotide level, at least one of their combinations performed at least as well as any single program in terms of sensitivity and specificity. The same was true at the exon level.

Allen and colleagues also compared several programs in two different ways and tested on sequences from *Arabidopsis thaliana* (Allen et al., 2004). The first combination of Genscan, GeneMark.hmm, and GlimmerM performed better than GeneMark.hmm, which performed the best out of all the individual gene finders on this test set. Likewise, a second combination of these three programs plus TwinScan and a newer version of GlimmerM performed significantly better than any one program.

The results of these combinations of gene finders are not surprising, given the fact that each program finds a different subset of genes. This should be an area of focus for those interested in improving gene prediction algorithms. It would be interesting to identify ways to quickly combine many or all existing programs trained for the same organism, and determine the upper limit of predictive power by combinations of programs.

Testing gene predictors on real sequence

Presented below is a true scenario in which accurate gene prediction is very important. In a situation in which a P-element insertion causes a certain phenotype (in this case, suppression of lethality due to overexpression of another gene), one would look for a gene close to the insertion site as the most likely candidate for the gene of interest. However, the gene nearest this P-element has two different mRNA models listed, according to FlyBase (Figure 2). In one, the P-element is inserted >5 kb upstream from the first exon. In the second model, the P-element insertion is less than 1 kilobase (kb) downstream from the first exon. Suppose an investigator would like to generate stronger alleles of this gene. Excision of the P-element is often imprecise

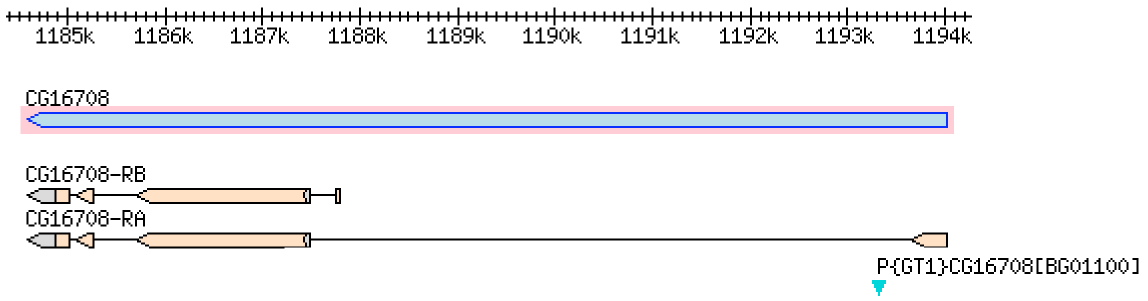

Figure 2. FlyBase model of CG16708 gene structure. Exons are represented by tan or grey boxes; P-element insertion site is indicated by upside-down cyan arrow.

and deletes a region surrounding the insertion site. In the first model, excision of the P-element is likely to only delete non-coding sequences. If these sequences were important, they would be regulatory regions and not coding regions, and would perhaps not be likely to generate stronger alleles than the original insertion. However, in the second model, the excision of the P-element is likely to delete portions of the gene coding for mRNA and would likely generate a stronger hypomorphic allele. The investigator's choice of whether or not to proceed with the P-element excision is dependent upon accurate gene modeling.

A ~20 kb region surrounding this P-element insertion site was used as test sequence for prediction by Fgenesh and GrailEXP (an updated version of the GRAIL program discussed above). Within this sequence, Fgenesh predicted 6 genes. Within the prediction, the exon structure of the gene corresponding to the position of CG16708 matched the first model above, missing the first exon close to the P-element insertion site. GRAIL, however, was unable to identify the first exon in either model. Instead, the program seemed to only identify protein-coding exons (note that, for either mRNA model given by FlyBase, the same protein sequence is predicted).

Both of these programs failed to answer the question of which gene model for CG16708 is correct. Perhaps other gene predictors would perform better, but it will be important to focus more on non-protein-coding exons rather than only protein-coding, as these regions perform important regulatory functions.

Works Cited:

Allen, J. E., Pertea, M. and Salzberg, S. L. (2004). Computational gene prediction using multiple sources of evidence. *Genome Res* **14**, 142-8.

Besemer, J. and Borodovsky, M. (2005). GeneMark: web software for gene finding in prokaryotes, eukaryotes and viruses. *Nucleic Acids Res* **33**, W451-4.

Borodovsky, M. and McIninch, J. (1993). GeneMark: Parallel Gene Recognition for Both DNA Strands. *Comput Chem* **17**, 123-33.

Burset, M. and Guigo, R. (1996). Evaluation of gene structure prediction programs. *Genomics* **34**, 353-67.

Lukashin, A. V. and Borodovsky, M. (1998). GeneMark.hmm: new solutions for gene finding. *Nucleic Acids Res* **26**, 1107-15.

Majoros, W. H., Pertea, M. and Salzberg, S. L. (2004). TigrScan and GlimmerHMM: two open source ab initio eukaryotic gene-finders. *Bioinformatics* **20**, 2878-9.

Mathe, C., Sagot, M. F., Schiex, T. and Rouze, P. (2002). Current methods of gene prediction, their strengths and weaknesses. *Nucleic Acids Res* **30**, 4103-17.

Pavlovic, V., Garg, A. and Kasif, S. (2002). A Bayesian framework for combining gene predictions. *Bioinformatics* **18**, 19-27.

Salamov, A. A. and Solovyev, V. V. (2000). Ab initio gene finding in Drosophila genomic DNA. *Genome Res* **10**, 516-22.

Salzberg, S. L., Delcher, A. L., Kasif, S. and White, O. (1998). Microbial gene identification using interpolated Markov models. *Nucleic Acids Res* **26**, 544-8.

Salzberg, S. L., Pertea, M., Delcher, A. L., Gardner, M. J. and Tettelin, H. (1999). Interpolated Markov models for eukaryotic gene finding. *Genomics* **59**, 24-31.

Uberbacher, E. C., Xu, Y. and Mural, R. J. (1996). Discovering and understanding genes in human DNA sequence using GRAIL. *Methods Enzymol* **266**, 259-81.

Fgenesh:
http://sun1.softberry.com/berry.phtml?topic=fgenesh&group=programs&subgroup=gfind

GRAILEXP:
http://compbio.ornl.gov/grailexp/