Integrating out model parameters in generative and discriminative classifiers

Niko Brümmer and Edward de Villiers
AGNITIO LABS, South Africa

May 2011

In this note we compare the general forms given by generative and discriminative probabilistic solutions for the basic classification problem in pattern recognition, where a supervised training database is available.

1 Problem definition

We are interested in the pattern recognition problem of inferring the class (or category) of an input pattern, given a supervised training database of example patterns which are annotated with the correct classes.

1.1 Notation

We denote the discrete set of classes as $\mathcal{C}$ and the space in which patterns live as $\mathcal{X}$.

We are given a supervised training database, denoted $D = (X, S)$, where $X = x_1, x_2, \dots, x_T \in \mathcal{X}$ are T input patterns and $S = c_1, c_2, \dots, c_T \in \mathcal{C}$ are the corresponding classes to which they belong.

The purpose of the whole exercise is: Infer the unknown category, $c' \in \mathcal{C}$, to which a new (test) pattern, $x' \in \mathcal{X}$, belongs.

We shall use x, c to generically refer to any training or test data.

We shall make extensive use of *graphical model* notation to represent conditional independence relationships via *Bayesian networks*. We recommend that the reader be familiar with how to apply the *d-separation* criterion to read conditional independence relationships from Bayesian network graphs.[1]

[1]See for example chapter 8 in C.M. Bishop, *Pattern Recognition and Machine Learning*, Springer 2006; or Tom Minka's tutorial at: `http://research.microsoft.com/en-us/um/people/minka/papers/diagrams.html`

1.2 Generative vs discriminative

We start by stating the basic modelling assumption that the data is *iid*, given some parametric probabilistic model. We shall apply this assumption to both generative and discriminative models. We denote our generic probabilistic model by $\mathcal{M}$. It describes the relationship between the patterns and the classes in the sense that it gives the joint pdf, $P(x, c|\mathcal{M})$, for any $x \in \mathcal{X}$ and $c \in \mathcal{C}$. The basic assumption is now that if $\mathcal{M}$ were known, then observing more data would not give any additional information not already contained in the model: $P(x, c|\mathcal{M}) = P(x, c|\mathcal{M}, Z)$, where Z is any collection of pairs from $\mathcal{X} \times \mathcal{C}$, which excludes (x, c). This makes the new data, (x', c'), conditionally independent of the training data D, if $\mathcal{M}$ were given:

$$P(S, X, x', c'|\mathcal{M}) = P(S, X|\mathcal{M})P(x', c'|\mathcal{M}) \tag{1}$$

and also gives: $P(X, S|\mathcal{M}) = \prod_{t=1}^{T} P(x_t, c_t|\mathcal{M})$. But $\mathcal{M}$ is not given, so the new data becomes dependent on the training data, because we effectively have to infer the model parameters from the training data.

The basic conditional independence assumption (1) can be represented via a *Bayesian network* as

$$\begin{tikzcd}
S \arrow[dd] & & c' \arrow[dd] & & S & & c' \\
& \mathcal{M} \arrow[ul] \arrow[ur] \arrow[dl] \arrow[dr] & & \text{or} & & \mathcal{M} \arrow[ul] \arrow[ur] \arrow[dl] \arrow[dr] & \\
X & & x' & & X \arrow[uu] & & x' \arrow[uu]
\end{tikzcd} \tag{2}$$

where the two forms are equivalent in the sense that both encode (1). The left form suggests a generative model, while the right suggests a discriminative model. By explicitly factorizing the model, we can obtain generative or discriminative model flavours.

For the *generative factorization*, we let $\mathcal{M} = (\theta, \pi)$ and:

$$P(x, c|\mathcal{M}) = P(x|c, \theta)P(c|\pi) \tag{3}$$

while for the *discriminative factorization*, we let $\mathcal{M} = (\lambda, \gamma)$ and:

$$P(x, c|\mathcal{M}) = P(c|x, \lambda)P(x|\gamma) \tag{4}$$

These two factorizations can be respectively represented as:

$$\begin{tikzcd} S & \pi \arrow[l] \arrow[r] & c' \arrow[d] \\ X \arrow[from=1-1, d] & \theta \arrow[l] \arrow[r] & x' \end{tikzcd} \quad \text{and} \quad \begin{tikzcd} S & \lambda \arrow[l] \arrow[r] & c' \\ X \arrow[u] & \gamma \arrow[l] \arrow[r] & x' \arrow[u] \end{tikzcd} \tag{5}$$

In the generative solution, the emphasis is on inferring the value of θ from the training data, while π may be simply given or otherwise trivially treated. In the discriminative solution, the emphasis is on inferring λ, while γ can be ignored, because when x' and X' are given, then all the other variables are independent of γ.

Finally note that if θ and π are given, we could in principle choose λ and γ to be equivalent in the sense that both forms would give the same joint pdf: $P(x, c|\theta, \gamma) = P(x, c|\lambda, \gamma)$. Then λ and γ would have to satisfy:

$$P(c|x, \lambda) = \frac{P(c|\pi)P(x|c, \theta)}{P(x|\gamma)}, \qquad P(x|\gamma) = \sum_{c \in \mathcal{C}} P(c|\pi)P(x|c, \theta) \tag{6}$$

In the next two sections, we discuss the generative and discriminative solutions in more detail.

2 Generative solutions

We reproduce the left-hand graph of (5):

$$\begin{tikzcd} S \arrow[d] & \Pi \arrow[d] & & \pi \arrow[d] \\ X & \theta \arrow[l] \arrow[r] & x' & c' \arrow[l] \end{tikzcd} \tag{7}$$

where we have: (i) omitted the arrow from π to S, because we will always treat S as given[2] and (ii) introduced the given hyperparameter Π, which defines a prior $P(\theta|\Pi)$ for the unknown model parameter θ. We consider π simply as given. The model can be summarized as:

$$P(X, \theta, x', c'|S, \Pi, \pi) = P(X|S, \theta)P(x'|c', \theta)P(\theta|\Pi)P(c'|\pi) \tag{8}$$

where $P(X|S, \theta) = \prod_{t=1}^{T} P(x_t|c_t, \theta)$.

[2]In the final problem, X and x' are also given, but we will need to write expressions of the form $P(X|\cdot)$ and $P(x'|\cdot)$, where these variables are temporarily considered unknown.

The purpose of the training database, $D = (X, S)$, is to help infer the unknown value of θ. The required end result of the inference is the posterior for the unknown class c', to which pattern x' belongs, given everything else: $P(c'|x', D, \Pi, \pi)$.

Below, we discuss three equivalent ways to express the exact *class posterior* $P(c'|x', D, \Pi, \pi)$. All three depend on the ability to compute *parameter posterior* distributions of the form $P(\theta|\cdot)$. In the first, we form the expectation of the *class likelihood* $P(x|c, \theta)$ w.r.t. $P(\theta|\cdot)$. In the second, we form the expectation of the class posterior $P(c|x, \theta)$. The third form is not an expectation.

Unfortunately, for most problems of interest, the posterior $P(\theta|\cdot)$ is analytically intractable and various approximations have to be used. Below we note only the simplest such approximation, known as the *MAP/ML plugin* solution.

2.1 Exact expected likelihood solution

The generative model gives what we refer to as the *likelihood*:

$$L(c|x, \theta) = kP(x|c, \theta) \tag{9}$$

where k is an irrelevant scale factor, which is *not* dependent on c, but which may depend on x. The required posterior can be expressed in terms of likelihoods in a straight-forward way by applying the basic rules of probability theory together with the conditional independence relationships as defined by (7), or (8):

$$P(c'|x', D, \Pi, \pi) = \frac{P(c'|\pi)\bar{L}(c'|x', D, \Pi)}{\sum_{c\in\mathcal{C}} P(c|\pi)\bar{L}(c|x', D, \Pi)} \tag{10}$$

where we have defined the *expected likelihood*:

$$\begin{aligned}\bar{L}(c|x, D, \Pi) &= k'P(x|c, D, \Pi) \\ &= k' \int_\Theta P(x|c, \theta)P(\theta|D, \Pi)\, d\theta \\ &= k'' \int_\Theta L(c|x, \theta)P(\theta|D, \Pi)\, d\theta\end{aligned} \tag{11}$$

where k' and k'' are scale factors which do not depend on c; and where $P(\theta|D, \Pi)$ is the posterior for the model parameter as inferred from the training database and where Θ is the support of $P(\theta|\Pi)$.

2.2 Exact expected posterior solution

Alternatively, we can compute the same answer, by *first* formulating the θ-dependent posterior:

$$P(c'|x',\theta,\pi) = \frac{P(c'|\pi)L(c'|x',\theta)}{\sum_{c\in\mathcal{C}} P(c|\pi)L(c|x',\theta)} \tag{12}$$

and *then* integrating out θ:

$$P(c'|x',D,\Pi,\pi) = \int_\Theta P(c'|x',\theta,\pi)P(\theta|x',D,\Pi,\pi)\,d\theta \tag{13}$$

In is interesting to note that the posterior $P(\theta|x',D,\Pi,\pi)$ that is needed here is conditioned also on the new datum x', whereas in (11) it was conditioned only on the training data, D.

It can be verified that (13) gives the same answer as (10), by noting that $P(\theta|x',D,\Pi,\pi) \propto P(\theta|D,\Pi)P(x'|\theta,\pi)$, where $P(x'|\theta,\pi)$ is the denominator of (12).

2.3 Exact plugin solution

Here we derive a third way to express the same solution. In the previous two solutions, there are two complications:

1. Computation of the posterior of the form: $P(\theta|\cdot)$. Unless $P(\theta|\Pi)$ is conjugate to the likelihood $P(X|S,\theta)$, the normalization constant for the posterior, $\int_\Theta P(\theta|\Pi)P(X|S,\theta)\,d\theta$, usually does not have an analytical solution.

2. The integrals (13) or (11), which are of the form $\int_\Theta f(\theta)P(\theta|\cdot)\,d\theta$, which (except for conjugate priors) are usually also analytically intractable.

The *exact plugin solution* (our terminology), avoids complication 2, but a properly normalized posterior $P(\theta|\cdot)$ is still required. This solution is obtained by factoring the joint pdf for the two unknowns, $P(c',\theta|x',D,\Pi,\pi)$, in two alternative ways and equating them:

$$P(c'|x',D,\Pi,\pi)P(\theta|c',x',D,\Pi) = P(c'|\theta,x',\pi)P(\theta|x',D,\Pi,\pi) \tag{14}$$

where we have used some of the conditional independencies encoded in (7) to remove unnecessary conditioning variables. Now we re-arrange and expand $P(\theta|x',D,\Pi,\pi)$ to find the required posterior:

$$P(c'|x',D,\Pi,\pi) = P(c'|\theta,x',\pi)\frac{\sum_{c\in\mathcal{C}} P(c|\pi)P(\theta|c,x',D,\Pi)}{P(\theta|c',x',D,\Pi)} \tag{15}$$

Several comments are in order:

- The LHS is independent of θ, while it appears in the RHS. This shows that the RHS is in fact *not* a function of θ and that therefore it can be computed by plugging in *any* convenient value, say $\hat{\theta}$, as long as the denominator remains non-zero.

- The computation of the *plugin posterior*, $P(c'|\hat{\theta}, x', \pi)$, is straight-forward, using (12).

- The computational challenge of this approach is to compute the parameter posterior $P(\theta|c, x', D, \Pi)$, for every $c \in \mathcal{C}$. The mechanics are the same as that of computing $P(\theta|D, \Pi)$, because we have effectively added another data point to the rest of the training data. But since this posterior is dependent on the new datum x', these posteriors must be evaluated for *every* new input.

- As promised above, the integral of the form $\int_\Theta f(\theta)P(\theta|\cdot)\,d\theta$ has been avoided.

2.4 Approximate MAP/ML plugin solution

The above three equivalent solutions are *fully Bayesian*, in the sense that the nuisance parameter[3] θ is integrated out, rather than making a point-estimate of it. The problem with the fully Bayesian solution is that the parameter posterior and associated integrals may be analytically intractable.

Various sophisticated approximation methodologies exist, but here we discuss only the simplest solution, where a *point estimate* of the value of θ is plugged into the θ-dependent class posterior. This is motivated by pretending that the parameter posterior approximates[4] an impulse at some value $\hat{\theta}$:

$$P(\theta|D, \Pi) \approx \delta(\theta - \hat{\theta}) \tag{16}$$

so that

$$\begin{aligned}\bar{L}(c|x, D, \Pi) &= k'' \int_\Theta L(c|x, \theta)P(\theta|D, \Pi)\,d\theta \\ &\approx k'' \int_\Theta L(c|x, \theta)\delta(\theta - \hat{\theta})\,d\theta = L(c|x, \hat{\theta})\end{aligned} \tag{17}$$

[3]A nuisance parameter is an unknown value, which is not the primary value of interest to be inferred.

[4]For some types of models and when enough training data is given, this may be a good approximation. However, the plugin solution is often used even if these conditions do not hold. See also the appendix.

and

$$P(c'|x', D, \Pi, \pi) \approx P(c'|x', \hat{\theta}, \pi) = \frac{P(c'|\pi)L(c'|x', \hat{\theta})}{\sum_{c\in\mathcal{C}} P(c|\pi)L(c|x', \hat{\theta})} \tag{18}$$

Below we discuss two well known ways to estimate the location of the impulse, $\hat{\theta}$.

2.4.1 MAP point estimate

In the *MAP point-estimate* solution, the location of the impulse is found by maximization of the posterior:

$$\begin{aligned}\theta_{\text{MAP}} &= \arg\max_{\theta\in\Theta} P(\theta|D, \Pi) \\ &= \arg\max_{\theta\in\Theta} P(\theta|\Pi)P(X|S, \theta) \\ &= \arg\max_{\theta\in\Theta} P(\theta|\Pi)\prod_{t=1}^{T} P(x_t|s_t, \theta)\end{aligned} \tag{19}$$

The simplicity of this solution lies in the fact that we don't have to perform any integrals over θ and we don't have to find the (usually) intractable normalization constant for the posterior $P(\theta|D, \Pi)$.

2.4.2 ML point estimate

As long as we pretend that the posterior $P(\theta|D, \Pi) \propto P(X|S, \theta)P(\theta|\Pi)$ is an impulse, then if the prior $P(\theta|\Pi)$ is smooth, it plays no role in determining the location of the impulse and can be ignored.

$$\theta_{\text{ML}} = \arg\max_{\theta\in\Theta} P(X|S, \theta) \tag{20}$$

Of course, in reality, $\theta_{\text{ML}} \neq \theta_{\text{MAP}}$.

For some further discussion of the applicability of the MAP/ML plugin solution, see the appendix.

2.4.3 Alternative plugin solution?

Examining (13), one may be tempted to form a different kind of MAP plugin solution by the approximation $P(\theta|x, D, \Pi, \pi) \approx \delta(\theta - \theta'_{\text{MAP}})$. This would give a different (and possibly more accurate) solution than (19). However, we cannot use the same trick here as in (19), where the normalization of the parameter posterior could be disregarded, because:

$$P(\theta|x,D,\Pi,\pi) = \sum_{c\in\mathcal{C}} P(c|\pi)P(\theta|x,c,D,\Pi) \tag{21}$$

in which the different terms have different normalization constants, which cannot be ignored during maximization of θ. In fact, if we had been able to compute $P(\theta|x,c,D,\Pi)$, then we might just as well have used it to directly compute the exact solution (15).

3 Discriminative solutions

Now we examine the alternate factorization of the model: $\mathcal{M} = (\lambda,\gamma)$. We repeat the right-hand graph of (5):

$$\begin{tikzcd} S & \lambda \arrow[l] \arrow[r] & c' \\ X \arrow[u] & \Pi' \arrow[u] & x' \arrow[u] \end{tikzcd} \tag{22}$$

where: (i) We have omitted γ, because in all our calculations X and x' will always be given. (ii) We have added the given hyperparameter Π' to form a prior $P(\lambda|\Pi')$ for the unknown model parameter. This model can be summarized as:

$$P(S,c',\lambda|x',X,\Pi') = P(S|X,\lambda)P(c'|x',\lambda)P(\lambda|\Pi') \tag{23}$$

where $P(S|X,\lambda) = \prod_{t=1}^{T} P(c_t|x_t,\lambda)$. The object of the exercise is to either approximate, or compute exactly, the fully Bayesian solution: $P(c'|x',D,\Pi')$.

Below we discuss two exact solutions and the approximate MAP/ML plugin solution.

3.1 Exact plugin solution

Again, we equate two alternative factorizations of $P(c',\lambda|x',D,\Pi')$:

$$P(c'|x',D,\Pi')P(\lambda|c',x',D,\Pi') = P(\lambda|D,\Pi')P(c'|\lambda,x') \tag{24}$$

where we have removed unnecessary conditioning terms. Rearranging gives:

$$P(c'|x',D,\Pi') = P(c'|\hat{\lambda},x')\frac{P(\hat{\lambda}|D,\Pi')}{P(\hat{\lambda}|c',x',D,\Pi')} \tag{25}$$

where once again, we can use any convenient $\hat{\lambda}$ as long as the denominator is non-zero. As before, the challenge is to compute properly normalized parameter posteriors, but we avoid further integrals w.r.t. λ.

3.2 Exact expected posterior solution

Reading conditional independence relationships from (22), we can express the desired class posterior as the expected value of the λ-dependent posterior, $P(c'|\lambda, x')$:

$$P(c'|x', D, \Pi') = \int_\Lambda P(c'|\lambda, x')P(\lambda|D, \Pi')\, d\lambda \tag{26}$$

where the expectation is w.r.t. the parameter posterior $P(\lambda|D, \Pi')$ and where Λ is the support of $P(\lambda|\Pi')$.

Compare (26) to (13), which is very similar. The discriminative solution appears simpler, because the parameter posterior is not conditioned on the new datum x'.

3.3 Approximate MAP/ML plugin solution

If we approximate: $P(\lambda|D, \Pi') = \delta(\lambda - \hat{\lambda})$, then:

$$\begin{aligned} P(c'|x', D, \Pi') &= \int_\Lambda P(c'|\lambda, x')P(\lambda|D, \Pi')\, d\lambda \\ &\approx \int_\Lambda P(c'|\lambda, x')\delta(\lambda - \hat{\lambda})\, d\lambda = P(c'|\hat{\lambda}, x') \end{aligned} \tag{27}$$

Again, seeking the maximum to find the location of the impulse, we can use MAP or ML solutions for $\hat{\lambda}$:

$$\begin{aligned} \lambda_{\text{MAP}} &= \arg\max_{\lambda\in\Lambda} P(\lambda|S, X, \Pi') \\ \lambda_{\text{MAP}} &= \arg\max_{\lambda\in\Lambda} P(S|\lambda, X)P(\lambda|\Pi') \\ \lambda_{\text{MAP}} &= \arg\max_{\lambda\in\Lambda} \log P(S|\lambda, X) + \log P(\lambda|\Pi') \\ \lambda_{\text{MAP}} &= \arg\min_{\lambda\in\Lambda} \sum_{t=1}^{T} - \log P(c_t|\lambda, x_t) - \log P(\lambda|\Pi') \end{aligned} \tag{28}$$

or

$$\lambda_{\text{ML}} = \arg\min_{\lambda\in\Lambda} \sum_{t=1}^{T} - \log P(c_t|\lambda, x_t) \tag{29}$$

Now notice that λ_{ML} is just the well-known discriminative training solution that would be found by minimizing the *cross-entropy*, or *logarithmic cost* criterion over the training database. Likewise, λ_{MAP} is a *regularized* version, with regularization penalty $-\log P(\lambda|\Pi)$.

4 Conclusion

Generative and discriminative solutions are just two different ways to formulate models for the joint distribution for class and pattern. Although we did not show this in detail, exact solutions via the two different routes can in principle give the same answers, i.e. $P(c'|x', D, \Pi, \pi) = P(c'|x', D, \Pi')$, provided the models and priors agree.

Generative and discriminative solutions will however give different answers when the MAP/ML plugin approximations are used. The question of which plugin method is better, depends on the details.

4.1 Questions

We end by posing two as yet unanswered questions, which were inspired by this document.

- The discriminative plugin approximation (i.e. cross-entropy minimization) is widely believed to give superior accuracy to the generative MAP/ML likelihood plugin approximation, when the form of the generative model poorly represents the data. Do the fully Bayesian generative and discriminative solutions also have this property? Are they also robust to inadequate modelling assumptions?

- The *exact plugin* (our terminology) solutions of sections 2.3 and 3.1, seem to be unknown in the literature. Could they give any real advantage over the more traditional approaches of sections 2.1 and 3.2? One often sees in the literature two approximation steps, the first to approximate the parameter posterior and the second to approximate the integral over the approximate parameter posterior. Our formulation suggests that perhaps only the first approximation step is necessary. Are there new stochastic or deterministic approximations that can be inspired by this formulation of the problem?

5 Appendix: Generalization of applicability of plugin approximation

The condition $P(\theta|D,\Pi) \approx \delta(\theta-\hat{\theta})$ is an unnecessarily restrictive assumption for motivating the approximate MAP/ML likelihood plugin solution of section 2.4. There are more general conditions under which such plugin solutions can give accurate approximations. We demonstrate this via an example.

5.1 Mixture model example

Consider, for example, the case where $\theta \in \Theta$ parametrizes a mixture model of some kind. The mixture model parameters are *non-identifiable* in the sense that any permutation of the mixture labels will give an equivalent model. Let $\mathcal{I}$ be the set of all the possible ways to permute the mixture labels and let $\theta^i \in \Theta$ denote the model parameters under permutation $i \in \mathcal{I}$. Since the mixture label permutation changes nothing essential about the model, we have $P(c,x|\theta^i,\pi) = P(c,x|\theta^j,\pi)$, for every $c \in \mathcal{C}$, $x \in \mathcal{X}$ and $i,j \in \mathcal{I}$. This also gives: $L(c|x,\theta^i) = L(c|x,\theta^j)$. If we choose our parameter prior to also be indifferent to the permutation, so that $P(\theta^i|\Pi) = P(\theta^j|\Pi)$, then the parameter posterior will also have this property: $P(\theta^i|D,\Pi) = P(\theta^j|D,\Pi)$.

Now, suppose that when there is enough training data, the parameter posterior becomes concentrated as a sum of impulses, of the form:

$$P(\theta|D,\Pi) \approx \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \delta(\theta-\hat{\theta}^i) \tag{30}$$

where $|\mathcal{I}|$ is the number of permutations; and where all the impulse locations, $\hat{\theta}^i$, are permutations of each other. Using (30) in (11), we can now approximate the expected likelihood as:

$$\begin{aligned}\bar{L}(c|x,D,\Pi) &= k'' \int_\Theta L(c|x,\theta)P(\theta|D,\Pi)\,d\theta \\ &\approx \frac{k''}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} L(c|x,\hat{\theta}^i) = \frac{k''}{|\mathcal{I}|}L(c|x,\hat{\theta}^\ell)\end{aligned} \tag{31}$$

where $\ell \in \mathcal{I}$ is any permutation. The constant $\frac{k''}{|\mathcal{I}|}$ is unimportant, because any likelihood scale constant cancels in (10). In our maximization step (MAP or ML), we can now seek any of the equivalent maxima $\hat{\theta}^\ell$.