Wine Cellar: An OCR Based Wine Recommendation Application For The Android Mobile OS

ABSTRACT

Wine Cellar is a search engine application developed for the Android Operating System’s Donut (1.6) build. The program features a search system focused on the collective display of results, rather than individual queries. This allows for a user to analyze all his options at the same time, in order to make a more informed decision. In addition, the application has the ability to perform the fore mentioned queries from an image input from the mobile device, as a more comprehensive and faster input method. On a basic level, Wine Cellar features a standard text-based search engine to perform its collective queries, but the main feature lies within the application’s ability to use the mobile device’s built-in camera to capture an image and use it as an input for the query, by applying an OCR algorithm to the captured image.

Existing wine review applications for mobile phones provide functionality ranging from simple lookup of wine brands and types to recommendations to the user and management of wine collections. The offered services, sizes of the databases and sources of information change across developers and mobile operating systems, but all these programs use text entry as the input method for the query.

The intent is to provide the user with a more advanced interface to the information related to the beverages offered at a certain location. This is accomplished through the integration of a text-recognition algorithm applied to images from the phone’s camera. Once an image of the wine menu is obtained, the program extracts the text from it and use it as the input for the search. This query will be specialized on the choices at the location and will return a comprehensive evaluation of the choices to help the user make a decision.

By using this approach, the hope is to allow for a more practical and universal lookup method, as well as to include the capability of focusing the search on the specific wine list of a restaurant.

The approach taken includes a server running a adaptation of an Optical Character Recognition (OCR) library called GOCR, to detect and extract text from images. By training the library to recognize specific kinds of fonts, This application runs in a server, which Android can send camera images to. The server will take care of the text recognition, keyword extraction, and database querying, and will report the results back to the mobile client application.

1. INTRODUCTION

Several related applications for mobile phones focus on providing a method to search through beverage databases, whether custom-built by the developer or external. In addition, different functions such as the display of prices, recommendations, and user ratings are included in some of these programs. Considering this, it is obvious that there exists a wide range of options for a user in need for wine descriptions or reviews.

While the base functionality of many of these applications may be desirable for this project, taking into account current cell phone technology, there could be improvements in the way the user inputs the query to the system. Specifically, the fact that virtually all modern mobile phones have a camera, in conjunction with the realization that the majority of mobile operating systems’ Application Programming Interfaces (APIs) provide access to the images captured by the phone, suggest that one could use the image of the menu at a restaurant as the input for the search.

1.1 The Present of Optical Character Recognition

Optical Character Recognition (OCR) is the process of automatic recognition of characters by computers in optically scanned and digitized pages of text. It has been studied for about fifty years, and it has progressed to a level where it can be utilized in user-end technology applications.

More generally, Document Analysis and Recognition (DAR) is a vast field and it has a great amount of real-life applications and, among its branches, character recognition has been considered as the one wit the most prominent future, and research efforts have predominantly focused on it. Generally, the study of character recognition has been approached from two distinct points of view. First, there is offline handwriting recognition, which refers to the process of recognition of words that have been scanned from a surface and have already been stored in a digital format. On the other hand is online handwriting recognition, where the writing is usually captured digitally in a direct fashion and stored digitally since its origin through various methods, such as stylus devices [?].

Due to the clarity and pure digital nature of the online handwriting method, it has admittedly been considered to achieve much better character recognition results than offline methods. In addition, digital input has the capability to provide more information about the text being written, such as the direction and order of the strokes, all of
which can, in turn, be analyzed and compared with previously known character patterns. On the other hand, scanned documents can have widely variable levels of quality and legibility, something crucial to the success of the chosen recognition method [7].

In the present day, character recognition has found application in data and word processing fields, such as acquisition of bank checks or processing of data records. Additionally, combined with different technologies like speech synthesis, it has been used as an aid for people who are visually handicapped. Today, Optical Character Recognition systems have been widely implemented for English, Chinese/Japanese, and handwritten numerals.

Text recognition methods applied on images have been studied since several years ago. Jain and Yu [9], although not focusing on ultimate character recognition, studied a method to extract text-only regions from images. In more modern methods, like the one described by Chen et al. [8], one can, in addition to isolating these regions, use character recognition on them to get the ultimate information needed from the image.

Even though the field of character recognition has seen major development and progress in the past half century, the issue of quality of the image which is used as a source remains the most prominent obstacle in the way towards optimally accurate systems that can be used in every day situations.

1.2 OCR Libraries

There exist several low-level programming libraries with the focus on aiding the development of Optical Character Recognition applications. Most of them are built to provide developers with ease of access to basic functions and tasks that are common to most OCR applications. Some of the most prominent Optical Character Recognition libraries today are described below.

1.2.1 GOCR

GOCR [7] comes packed in a library called libgocr. This library’s purpose is to provide access to low-level functions that OCR applications should not, ideally, have to focus on. These functions include several I/O methods, feature-rich linked list objects and more advanced functions that perform steps that are usual in almost any character recognition application, such as training the engine with a group of sample images or processing an input image based on the inner engine.

Some of the main features of the libgocr library include:

- File input for common image types and Unicode support
- Modular code to allow for partial implementations (plug ins) that integrate with current code
- Linked lists, hash tables and other objects to handle data

1.2.2 Tesseract

Tesseract OCR library [?] rated amongst the first three engines in he 1995 UNLV Accuracy test. It provides code to read a binary image and output text. It works at a higher level than its GORC counterpart, providing a full character recognition background system. More specifically, a TIFF (Tagged Image File Format) reader is built-in to read uncompressed TIFF images, which allows the engine to receive input from a captured image of text.

Tesseract features support for Linux, Windows, and Macintosh platforms, and there has been some success with Cygwin builds as well. In addition it includes support for several languages, with Chinese coming shortly.

However, the project has been inactive between 1995 and 2006 and in recent years so its inner functionalities are not optimally updated and maintained.

1.2.3 OCRAD

The GNU Ocrad project [?] is an OCR library which is based in a feature extraction method. Its input formats range from pbm (bitmap), pgm (grayscale) and ppm (color) formats and returns text that the engine detects.

This engine, specifically, has the added functionality of a layout analyzer. This portion of the engine was implemented with the idea of identifying regions of an image where text is located, such as columns or blocks on printed pages.

1.3 Wine Cellar’s Approach using OCR

The improvements that this approach would provide to current applications are plenty. By eliminating the need to type, the user would have a much easier way to provide the program with the information needed for the query. From a usability point of view, this is an enhancement that would draw more users in, as well as allow the customers at restaurants to efficiently and quickly evaluate the types of wine available to them. If, in addition to this technological improvement, a unified, comprehensive display of the results specific to the menu in question is implemented, the functionality and appeal of this kind of applications would dramatically increase.

2. RELATED WORK

Without a doubt, the platform with the most wine review applications is the iPhone. Several developers have implemented review searches in general and, more relevantly, several others provide with information aimed specifically at users using the application at the moment of choice, that is, when looking at the drink menu in a restaurant.

Nirvino’s Wine Ratings Guide2, for example, offers wine ratings, recommendations and prices. Some other applications, like Wine Enthusiast Guide3 give the user advice from wine experts, as well as a more detailed description of each kind of drink.

For the presentation of the query results in our specific application, some aspects of Wine Ratings Guide’s model could be used. The flavors profile is something useful to the end user as an objective description of the wine, and the pairings of each kind of beverage with certain types of food could also be a good guide for a consumer at a restaurant, specifically. On the other hand, Wine Enthusiast’s approach by having experts - as opposed to regular consumers - review the different kinds of wine provides, perhaps, a more insightful explanation of the wine’s taste. A combination of these two implementation choices, in addition to others, could help achieve a presentation of results that is both easy.

2 ©2007-2009 Nirvino LLC
3 mobileAge, LLC
to look at and useful to be able to choose among all the available options.

According to studies [?], given the knowledge of the existence of reviews on a certain product, consumers are likely to look at them and take them into consideration to make their final choice, or at least for reassurance. The review model of the application would have to be able to provide a deep analysis based on the available choices, the user’s preference, and the previous consumers’ experiences, while maintaining the objectivity of the results.

On the other hand, the main purpose of this application is to be able to compare the different characteristics of all the options in the menu. This suggests, intuitively, a display of the result that focuses on a comparative view of the different available types of wine, as opposed to the features of each individual one. Existing applications don’t usually include this capability, or don’t put enough emphasis on it.

### 2.1 Android Applications

The Android Mobile Operating System has been chosen for development, mostly due to its relatively smaller market for applications, and the more limited amount of programs similar to Wine Cellar. In this platform, there aren’t any services focused on providing wine reviews. However, there has been some success with text recognition, in conjunction with keyword extraction, used for product search over the internet using the mobile phone’s built-in camera instead of user input.

The most relevant example of this is AlchemySnap 4, an application for the Android Mobile Platform implementing OCR to search for products online. AlchemySnap integrates search results from sources like Google, Wikipedia and Twitter from a query carried out by inputting an image taken with the Android phone’s camera (Fig. ??). This application uses the AlchemyAPI to perform an entity search on the recognized text. This algorithm extracts relevant terms from text and detects the parts of the input that should be transferred as query keywords (Fig. ??).

While the general use and purpose of the application would be similar to that of the existing implementations in the Android Operating System, as well as other mobile platforms, the most prominent enhancement would be in the way the query terms are passed on to the search system. All current applications use a classic search model where the user inputs the name - or part of the name - of the wine and the system returns a result or list of results accompanied with ratings, reviews, and/or characteristics of the requested type of drink. This is good enough at a certain level, but, as mentioned before, it is restricted to searching for one individual kind of wine at a certain time. This fact alone requires the user to repeatedly input queries if he wants to look at the features of each one of the options in a menu.

With a text recognition input system, however, not only can the user save search time by pointing the camera at the entire ensemble of choices in the menu, as opposed to a specific one, but the results can be optimized to give information that is completely focused on a comparative analysis of the different options.

4 ©AlchemyAPI by Orchestr8 - http://www.alchemyapi.com/tools/alchemy SNAP/
clear user interface, aiming to be a helpful, concise aid for the customer to choose well.

3.3 Input Method

In terms of the input technology, implementing text-recognition technology in a wine review application allows for several usage advantages. From the user’s standpoint, choosing one drink among many available options through one of the current applications necessarily requires the individual search of each choice in the menu. If we allow an action as simple as pointing the camera at the menu as the input method for the software, we eliminate most of the time spent in navigating back to the search and re-typing kinds of drinks. Considering, especially, the predicted circumstances the user will find himself in when using the program, namely having choose a drink in a couple minutes, this is a really considerable advantage from the point of view of usability.

The other main advantage of this input method is the fact that, due to the ability to collect several wine names from one single image, we can generate search results that already include information on every type of drink in the menu. The simultaneous possession of the names of all the wines the user needs to know about, gives the software the chance to perform a more detailed and useful analysis of the choices. If, for example the program includes a user profile, the characteristics of all the wines available could be compared to the user preferences to provide accurate recommendations in nearly real time.

While the methods presented in these articles are useful, a much simplified character recognition method is used for this specific application. Due to the fact that one can, in general, assume that pictures of wine menus will consist of mostly a background without many distracting elements - that is a white or nearly white background -, it is important to focus on improving the contrast between the written words and the clear background. To improve the accuracy of the text detection, the input image can be enhanced in real time to emphasize the text on it. Using this technique, the text recognition algorithm can make assumptions about the image to look for text in a mostly clean background, thus making the process much quicker.

4. SYSTEM IMPLEMENTATION

As outlined before, the design makes use of an Optical Character Recognition library adapted for Wine Cellar, and relevant fonts have been chosen to train the Recognition engine to be able to identify these specific kinds of typography in the various images to be used as input. This library is the main component of the current development, and its components, functions and usage are described in this section.

4.1 OCR

The OCR engine used is written for the Java programming language. This language has been chosen for development due to its close relationship to the Android development environment, and the convenience of implementation for servlets to be used for text extraction from images. This specific OCR API is adequate because of its simplicity, and the ease of access it provides for custom applications, as is necessary in this case.

For the OCR engine to work, there has to be one or more prepared images containing training character ranges for the application, as illustrated in Fig. ???. These images, with the desired fonts to be detected, are inputted to the library, along with information about the characters contained in them. This information is used by the library, and it is stored in a scanner object to enable it to recognize similar text in future inputted images.

Once the training process reaches its completion, the API allows for the use of the trained image scanner to analyze input images. The image that one wishes to analyze is passed to the scanner object first. The scanner reads this image from the filesystem into a Java image object, and proceeds to study the image and detects the characters it was trained to find. These found characters care internally highlighted - as illustrated in Fig. ?? - and later outputted as the search result to the analysis.

This output is what the server uses for the query, and the results are forwarded back to the mobile client application.

4.1.1 Text Recognition

Considering the Java OCR engine used for this application, it is important to select a good variety of fonts to train the character scanner in advance and allow it to recognize many different types of input from wine menus. In general, the most used kinds of typography in restaurants include some popular Serif fonts, such as Times New Roman and Garamond, as well as Sans-Serif fonts like Verdana, and more rounded versions of these, as illustrated in the following paragraphs.

For Wine Cellar, the OCR Library has been adapted for the required fonts. Different examples of this are illustrated in the Appendix. In Fig. ??, the font Times New Roman is loaded into the OCR Scanner object during the training process. Next, the OCR implementation recognizes the characters in the image, and outputs the results to the terminal for further utilization and analysis (Fig. ??).

As seen in this example, the recognition algorithm accurately identifies the main features of the wine menu entry, such as name, kind, year, and price. These are the important pieces of information needed for a successful analysis, query, and result obtention.

Another demonstration, involving a vastly different family of fonts, illustrates the power of the training process. By inputting both a set of Serif characters and a set of rounded characters into the OCR engine’s training process, one can broaden the range of different fonts that the application can identify.

An example menu that uses the rounded typeface “Heiti SC Light” is exemplified in Fig ???. In this example, the same trained OCR Scanner utilized previously is used, with the same training process, and, as expected, the application identifies the characters and outputs the results to the terminal (Fig. ??).

These figures illustrate the versatility of the Java OCR engine used in Wine Cellar, and the degree of success when extracting basic text from images after the necessary training stage.

Due to the efficiency of the base OCR engine, Wine Cellar uses additional OCR servers to perform a cross-referenced query with the same input image and thus obtain more accurate results. Since each of these servers represents a different OCR implementation and uses a different library, the results vary and, together, provide an improved result for the recognition of characters in the image.
4.1.2 Wine Cellar’s interaction with OCR server

In terms of Optical Character Recognition, the role of the client Wine Cellar application, running on the Android mobile device is to capture an image from the camera (Fig. ??), send it to the OCR servers and receive the recognition results back.

To implement these functions the client application was built to take the image file captured from the built-in camera and send it to the OCR server as a POST request. For this purpose, the Android application makes use of the Java HTTPClient library ⁵ to create package that uploads the image file to the server via the multipart/text HTTP form protocol.

Considering the need for the speed for the Wine Cellar client application, and the fact that the OCR servers perform both poorly and slowly on files larger than 1MB, the mobile application performs an extra step after capturing the image where the file is further compressed to allow for fast performance of the text recognition process and more responsiveness for the user. It was found that image compression did not significantly decrease the accuracy of the character recognition and, in some cases it helped the process. Due to this fact, the compression step is an important part of the client program.

4.2 Android User Interface

The user interface that interacts with the server and the Snooth database was built for the mobile device. As mentioned previously, this client application was developed for the Android Mobile Operating System, and is in charge of obtaining the image from the phone camera, transferring it to the server and obtaining the search results from it when they are available.

The main goal for the user interface is to provide a concise, fast and stylish way to send images and display the results of the query. Even though Snooth provides a great amount of information related to each kind of wine inputted to the search, the client cleans these results and relate the wines to each other to provide a better recommendation to the user based on his preferences.

The descriptions for each wine are initially short and simple, the user has the option to further inquire about the nature of the various beverages, based on the information returned from the Snooth database.

4.2.1 Database Querying

Once the OCR process was fine-tuned to extract the relevant text from the input images, the next step in development was to perform a simple query to a wine database to obtain descriptions, opinions and recommendations related to each kind of wine detected from the user search.

On the interface side, one has to be careful to recognize the influence of the displayed information on the consumer, and specifically the expectation that the style of presentation of the characteristics of the wine will generate. Studies ⁷ show that the ultimate opinion of the user about the product is heavily worsened when an excessively high expectation is generated in advance. Since there is a direct correlation between the user’s opinion of the chosen wine and the image of the quality of our software’s information and recommendations, it is essential to optimize the display of the different aspects of the wine in a concise and unbiased manner.

With this in mind, the main source chosen is Snooth ⁶. This vast wine database encompasses the desired aspects for Wine Cellar, and its size and variety allows for the obtention of results for searches of a great variety of kinds of wine.

As illustrated in Fig. ??, Snooth provides basic information about the wine, including name, type, rating and origin. These parameters are useful to guarantee the correct mapping between search terms and desired wine types, and are good start to the obtention of accurate results.

Furthermore, Snooth offers a much broader detailed description of the wine, as exemplified in Fig. ???. This explanation includes both a short entry by the reviewers - which is important when determining the quality of the wine and the relationship with the user’s preferences - and a description of the wine by the maker. This, however biased, is a good insight for the user to be able to have knowledge of the production process of the wine. Additionally, the website recommends wines that are close in flavor and that users prefer. This is one of the main features desired for Wine Cellar as well.

The implementation of the searching feature was completely built into the client application on the mobile phone. Its main component consists of a Java library written for term extraction from the Snooth online database. First, the library obtains the search terms, either from the text input (Fig. ??) or from the OCR server returned results and creates an HTTP request for each kind of wine that has to be queried and transmits these search terms to the Snooth system. When the HTTP response arrives, the Search library performs a detailed analysis using regular expressions to extract the relevant content from the searches. To improve speed, this content is initially limited to the name of the wines resulting from the search only. However, the library provides the same functionality to extract the terms from the individual page for a wine, in order to be able to obtain each detailed piece of information about it.

4.2.2 Results

The results of the querying process are returned by the internal Wine Cellar Search engine to the main core of the mobile application. Once they are received, they are initially displayed to the user in a concise and simplified way (Fig. ??). This decision was made thinking of ease of access and fast decision-making at the moment of usage. With simple display, the user can easily click on different wine types to see more information about it.

Once the user chooses a wine to further explore, the Wine Cellar application, again, uses its Search library to perform a detailed term extraction query to the online database. Unlike the initial search for wine results, this search is performed far more in-depth, and includes many other aspects beyond the name of the wine such as year of production, winery and region of origin, price and rating and a detailed description by experts. The mobile application then displays all these new pieces of information in an organized way to let the user explore the item in depth (Fig. ??).

From this point, the user still has the choice to visit the full-fledged Snooth page for the wine by asking the mobile

⁵Part of the Apache Commons package - http://commons.apache.org/

⁶http://www.snooth.com
application to display more information about the selected beverage.

5. SYSTEM PERFORMANCE

There are a few parameters that can help evaluate the efficiency of the result. Firstly, we can compare the performance of the program with existing implementations by contrasting the time it takes to make a query for the wine list with the time it takes with other software to look up each wine individually.

5.1 Speed

Compared to other applications, mainly AlchemySnap, the Wine Cellar initial implementation of the Java OCR algorithm performs well. The speed with which the text results are reported back by the application is faster than AlchemySnap. However, one has to take into account that the current process of development for Wine Cellar up to this point only includes text recognition. One has to account for the time spent with the querying and reporting of results.

On the other hand, these extra steps are not nearly as computationally expensive as analyzing the input image from the camera and extracting the text from it using the API. Considering this part of the workflow is the most time consuming, Wine Cellar is within a reasonable margin of performance in relationship to similar existing applications.

5.2 Accuracy

In terms of accuracy, the Java OCR library, as illustrated in the previously mentioned figures, performs very favorably for good enough input and aided by a good selection of fonts to be inputted to the training stage of the search. If these character sets match the targeted typography styles with enough proximity, the result is very accurate, and very little correction needs to be applied to the extracted text for further use.

However, when the image quality is low, or none of the training fonts match the input image font with enough accuracy, the performance of the algorithm decreases and the extracted text contains undesired characters that could impact the accuracy of the query and therefore the results that are reported back to the user.

Most images originated in modern Android-enabled devices, provide quality that is of enough quality to be used with the Optical Character Recognition. However, there is a slight decrease in accuracy, as illustrated in Fig. ?? and Fig. ???. The test is mostly recognized by the Java OCR library. However, a small percentage of the characters are left out and white space is slightly misinterpreted.

The Java OCR Library is largely dependent on the training fonts, and thus it becomes a crucial point to choose a large amount of training typefaces to cover all the possible different inputs to the Scanner. However, optimization needs to be applied to obtain better results, specifically from phone generated images.

Furthermore, the quality of the camera device built into the mobile phone has an important effect in the quality of the captured image and therefore in the recognition process on the servers and the accuracy of the returned results.

6. REMAINING WORK

Text recognition has posed the largest challenge to the development process of Wine Cellar. Even when using several servers, detection of characters on images taken with an Android camera – especially using a device with a low resolution capture device – is not optimal. Post-processing is often not enough to obtain optimal results when performing a search from a camera image. The most important aspect to work on in the future is definitely the optimization of the recognition process.

6.1 OCR Optimization

Even though the Java OCR algorithm performs very well on input from good images where the training process has been carefully monitored, there are still some major disadvantages in the text detection for lower quality inputs, as well as typefaces with little or no spacing between characters.

These drawbacks have to be further addressed, given that the input images to Wine Cellar will certainly be of regular phone-camera quality. In the future development process, the first step is to solve this aspect, and modify the Optical Character recognition as much as needed to be able to deal with the expected quality of input and kinds of typography from wine menus.

6.2 Additional Features

In addition to these essential features aimed to provide a quick aid when choosing a wine type, the program can still be enhanced with other more advanced features, targeted mainly to cases where the user has sufficient time for a more detailed analysis of choices. Such capabilities could, for example, include the implementation of a recommendation algorithm that could single out the best wine choice, based on a comparison between the available options and previously defined user preferences. Additionally, the software could implement a full ranking of the menu choices in a similar way, or even ask for input from the user regarding his opinion on the selected wine, to add this information to the user profile and learn to make better suggestions every time.

Once an optimized implementation of text recognition and search, this model could conceivably be extended to other related fields. For example, one could use the same method to search for cheese recommendations, or even other kinds of products in a store or restaurant. Furthermore, a text recognition algorithm optimized for unfavorable image conditions could be useful in other applications, such as document scanning with low resolution cameras.

In terms of functionality of the software, the program can utilize the collected user’s opinion of the choice made by the program in order to qualify its own accuracy. Furthermore, additional feedback could be requested from the user, describing the accuracy of the recommendations. These pieces of information could be used to both evaluate and gradually improve the performance of the application.

7. CONCLUSION

Wine Cellar is a complete wine search application for the Android Operating System. Its internal search engine successfully retrieves wine information from the Snooth database, and its interface lets the user interact with this information in a concise way.
The input to the query can come from a text based input, similar to any other basic search engine. However, the relevant feature of the application is to implement Optical Character Recognition to allow the user to capture an image of a wine menu from the camera and use this image file as an input for the query. The accuracy of this method can allow for a successful search. While it depends on many factors and further optimization is necessary to obtain optimal results, the method still allows to obtain input and perform a search for found term and thus use a more intuitive method to inquire about types of wine.

The mobile application achieves the purpose of aiding customers to make a choice when at a restaurant, and the original addition of an OCR service for input makes of this application a useful way to perform a collective search for wine kinds to successfully make a satisfactory decision.

APPENDIX
A. INCLUDED FIGURES

Figure 1: AlchemySnap - Capturing Image

Figure 2: AlchemySnap - Extracted Text
Figure 3: Wine Cellar Workflow

Figure 4: OCR Library Demonstration - Training Character Range

Figure 5: OCR Library Demonstration - Text detection in image

Figure 6: OCR Library Demonstration - Results

Figure 7: OCR in Wine Cellar - Serif character recognition

Figure 8: OCR in Wine Cellar - Serif character recognition results

Figure 9: OCR in Wine Cellar - Round type character recognition

Figure 10: OCR in Wine Cellar - Round type character recognition results

Figure 11: OCR in Wine Cellar - Recognition on phone images

Figure 12: OCR in Wine Cellar - Recognition on phone images - Results
Folie a Deux Menage a Trois

![Image](image1.png)

**Figure 13: Snoop - Base information**

**Figure 14: Snoop - Description and recommendations**

**Figure 15: Wine Cellar - Search screen**

**Figure 16: Wine Cellar - Camera based search**

Zinfandel is considered America’s own great indigenous grape, even though its origins lie on the Adriatic coast. Planted throughout California and the Pacific Northwest, Zinfandel is at its best in warm regions with cooler temperatures during harvest. The wines can range from off-dry Rosés, White Zinfandel, and light bistro styled wines, to big, rich powerful wines – even luscious wines for dessert bottling. The flavors range from plummy to raspberry, although deep blackberry fruit and brambly spice tones are most common.

Winemakers Notes: Mnage Trois red brings together three strange bedfellows: Zinfandel, Merlot and Cabernet Sauvignon. To add complexity and retain character, each variety was fermented separately, then blended together prior to bottling. On top weve got Zinfandel, a saucy tease that brings blackberry and raspberry jam to the relationship. Merlot, with its generous mouth feel and red fruits, fits perfectly in the middle. The rich flavors and firm tannins of Cabernet Sauvignon make it the ideal candidate for the bottom layer of the trio. Cheese Pairings: 4-Year Gouda, GruyereFood Pairings: read
Merlot has gotten a bad rap as of late but there is no denying the wine's appeal. Packed with deep, dark berry fruit and nuanced hints of cocoa, herb and earth with a soft structure. Its velvety texture makes Merlot instantly appealing and an obvious alternative to similar, more tannic Bordeaux varieties.

More Info...