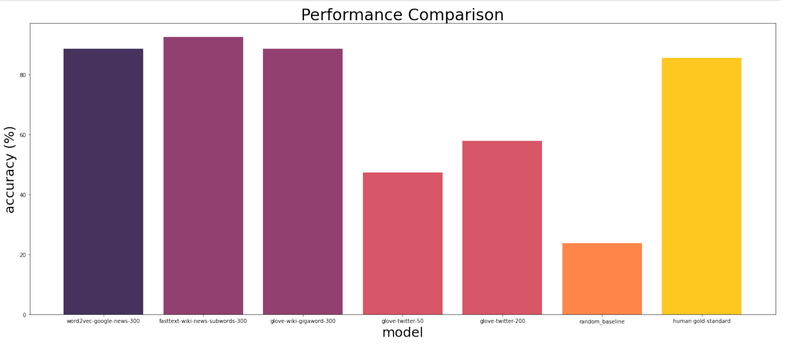
MP-03 Analysis

In [10]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [26]:
```
accuracies = np.array([0.88608, 0.925, 0.88608, 0.47368, 0.57895,  0.2375, 0.8557])*100
models = np.array(['word2vec-google-news-300', 'fasttext-wiki-news-subwords-300', 'glove-wiki-gigaword-300', 'glove-twitter-50', 'glove-twitter-2
```

In [43]:
```
plt.figure(figsize=(25,10))
plt.bar(models[0],accuracies[0], color='#46325c')
plt.bar(models[1:3],accuracies[1:3], color='#914070')
plt.bar(models[3:5],accuracies[3:5], color='#d75667')
plt.bar(models[5],accuracies[5], color='#ff8648')
plt.bar(models[6],accuracies[6], color='#ffc821')
plt.xlabel('model', size=25)
plt.ylabel('accuracy (%)', size=25)
plt.title('Performance Comparison', size=30);
```



Overview:
The bar graph above shows the accuracy of the five models as well as the accuracy of the random baseline and human gold standard provided by the students.

The worst performance comes from the **Twitter models**, while the best accuracy comes from **Wikipedia and Google's news**. These results are not surprising since the vocabulary used in the corpus of these models is very different. One often sees language that is more formal and grammatically correct in encyclopedias such as Wikipedia and newspapers such as Google news. While on the other hand, in social media sites such as Twitter, one would expect to see more informal and slang-like language.

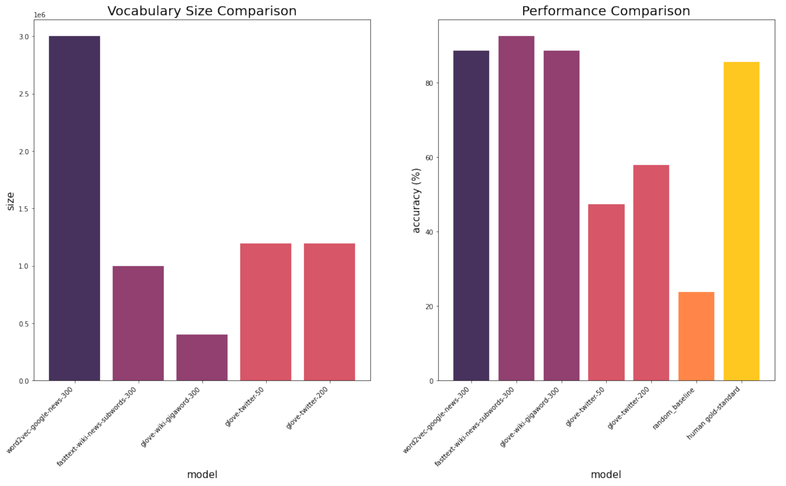
This difference in the language used in the models' corpus explains the difference in performance.

As for the random **baseline and human gold standard**, their results are rather self-explanatory. The accuracy of the random baseline is very low, as one would expect from a random answer generator. For the human gold standard, although I was not surprised to see it scoring well, I was surprised to see an artificial model do better than university students.

Comparing word embedding size:
Although all models with the highest word embedding size scored the best, we also see that the Twitter models, regardless of their word embedding size, did not perform great. This implies that there could be something beyond that, and we should perhaps look into the influence of the vocabulary size or type on the accuracy score of the model.

In [44]:
```
sizes = np.array([3000000, 999999, 400000, 1193514, 1193514])
```

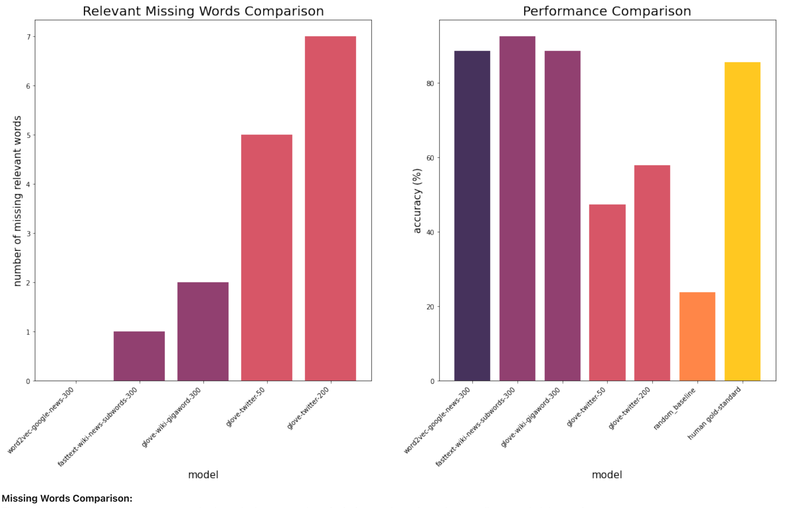
In [78]:
```
plt.figure(figsize=(20,10))
plt.subplot(1,2,1)
plt.bar(models[0],sizes[0], color='#46325c')
plt.bar(models[1:3],sizes[1:3], color='#914070')
plt.bar(models[3:5],sizes[3:5], color='#d75667')
plt.xlabel('model',size=15)
plt.xticks(rotation=45, ha='right')
plt.ylabel('size', size=15)
plt.title('Vocabulary Size Comparison', size=20)

plt.subplot(1,2,2)
plt.bar(models[0],accuracies[0], color='#46325c')
plt.bar(models[1:3],accuracies[1:3], color='#914070')
plt.bar(models[3:5],accuracies[3:5], color='#d75667')
plt.bar(models[5],accuracies[5], color='#ff8648')
plt.bar(models[6],accuracies[6], color='#ffc821')
plt.xlabel('model', size=15)
plt.xticks(rotation=45, ha='right')
plt.ylabel('accuracy (%)', size=15)
plt.title('Performance Comparison', size=20);
```



Comparing vocabulary sizes:
In the graphs above, we see that the second and third models, despite having the smallest vocabulary size, those models still have the best accuracy among all other models. This suggests that vocabulary size does not have a big influence on the performance of the model.

In [70]:
```
msg_words = np.array([0,1,2,5,7])
```

In [82]:
```
plt.figure(figsize=(20,10))
plt.subplot(1,2,1)
plt.bar(models[0],msg_words[0], color='#46325c')
plt.bar(models[1:3],msg_words[1:3], color='#914070')
plt.bar(models[3:5],msg_words[3:5], color='#d75667')
plt.xlabel('model', size=15)
plt.xticks(rotation=45,ha='right')
plt.ylabel('number of missing relevant words', size=15)
plt.title('Relevant Missing Words Comparison', size=20)
plt.subplot(1,2,2)
# plt.figure(figsize=(25,10))
plt.bar(models[0],accuracies[0], color='#46325c')
plt.bar(models[1:3],accuracies[1:3], color='#914070')
plt.bar(models[3:5],accuracies[3:5], color='#d75667')
plt.bar(models[5],accuracies[5], color='#ff8648')
plt.bar(models[6],accuracies[6], color='#ffc821')
plt.xlabel('model', size=15)
plt.xticks(rotation=45, ha='right')
plt.ylabel('accuracy (%)', size=15)
plt.title('Performance Comparison', size=20);
```



Missing Words Comparison:
The models with the lowest accuracy score also have the largest number of missing words, perhaps that contributes to their low performance.

Conclusion:
From the analysis above, it appears that the corpus has the biggest impact on the performance of the model.
More specifically:

1. **type of corpus:** what collection of texts were used to construct the model
2. **size of corpus:** how many texts and documents were used to build the model
3. **missing words:** words that are relevant to the synonym investigation are missing